Toward Robust Information Extraction Models for Multimedia Documents

Ali-Reza Ebadat

Thèse

THÈSE INSA Rennes
sous le sceau de l'Université Européenne de Bretagne
pour obtenir le grade de
DOCTEUR DE L'INSA DE RENNES
Spécialité : Informatique

présentée par
Ali Reza Ebadat
ÉCOLE DOCTORALE : MATISSE
LABORATOIRE : IRISA/INRIA

Toward Robust Information Extraction Models for Multimedia Documents

Thèse soutenue le 17 Octobre 2012
devant le jury composé de :

Emmanuel Morin
Professeur, université de Nantes / *Président*
Anne Vilnat
Professeur, université de Paris-Sud / *Rapporteur*
Patrice Bellot
Professeur, université d'Aix-Marseille / *Rapporteur*
Olivier Ferret
Chercheur, CEA LIST / *Examinateur*
Pascale Sebillot
Professeur, INSA de Rennes / *Directrice de thèse*
Vincent Claveau
Chercheur, IRISA - CNRS / *Co-encadrant*

Acknowledgement

This thesis would not have been done without the help of my colleagues, friends and family. I would like to thank my supervisor Prof. Pascale Sebillot for her great pieces of advice during this research. This thesis would not have been possible without the help, support and patience of my supervisor Dr. Vincent Claveau. My special thanks go to him for his wonderful pieces of advice in this thesis.

Besides my supervisors, I would like to thank the rest of my thesis committee: Prof. Anne Vilnat, Prof. Patrice Bellot, Prof. Emmanuel Morin and Dr. Olivier Ferret for their comments and questions.

I am grateful to Dr. Afshin Moein, Mihir Jain and Dr. Anh-Phuong Ta for providing me effective feedback on the thesis manuscript.

Last but not the least, I would like to send my greatest thank to my wonderful wife Shaghayegh and my sweet son Sam for their warm supports and patience.

Contents

Part I

Contributions à l'extraction d'informations robuste pour les documents multimédias

Chapitre 1

Introduction

Au cours de la dernière décennie, d'énormes quantités de documents multimédias ont été créées (par exemple, 72 heures de vidéos sont téléchargées sur YouTube chaque minute). Il est donc important de trouver un moyen de gérer ces données, non pas d'un point de vue simplement technique, mais aussi sémantique. Toute approche visant à faciliter ce processus nécessite d'avoir un aperçu du contenu des documents. Il existe principalement deux familles d'approches pour obtenir un tel aperçu des documents multimédias : soit par l'extraction d'informations à partir du document, soit en utilisant des données textuelles connexes provenant de sources externes (comme le Web). Dans la première famille d'approches, les diverses sources d'information internes aux documents, telles que la séquence de prises de vue vidéo [Sivic and Zisserman, 2003], le texte incrusté [Jung, 2004; Elagouni et al., 2012] et le contenu audio [Lawto et al., 2011] sont utilisés (voir [Hu et al., 2011] pour plus d'informations). L'extraction des objets tels que les voitures, les maisons, etc., des vidéos est cependant encore limitée aux scènes simples ayant un ou deux objets. La détection et l'extraction d'événements se révèlent par ailleurs encore plus difficiles. Dans la seconde famille d'approches, certains chercheurs ont proposé des modèles utilisant des ressources textuelles externes pour aider la gestion des documents multimédias [Buitelaar et al., 2008b; Reidsma et al., 2003].

1.1 Objectifs de la thèse

Le travail présenté dans cette thèse se situe à la croisée de ces deux approches. Il prend source dans un cadre plus large concerné par l'indexation de documents multimédia, puisqu'il a été mené dans le projet INRIA / TexMex[1] et a été fi-

[1] TexMex est une équipe de recherche s'intéressant à l'indexation de documents multimédia ; www.irisa.fr/texmex .

nancé par le projet Quaero[2]. Dans ce contexte d'indexation multimédia, toutes informations linguistiques ou textuelles relatives au document à traiter est un apport très précieux pour décrire son contenu. Dans les vidéos, par exemple, il peut s'agir de textes incrustés, des sites Web connexes, ou plus communément, des transcriptions des discours. À titre d'exemple, supposons qu'un utilisateur cherche les informations suivantes dans une banque de vidéos de matchs de foot :

- Toutes les scènes de but du match entre l'Allemagne et l'Italie pendant l'Euro2012.
- Toutes les scènes de fautes aboutissant à un carton jaune pendant l'Euro2012.

Pour répondre à ces besoins, une annotation sémantique des matchs est nécessaire ; et pour ce faire, une étape d'extraction d'informations, par exemple dans des textes décrivant les matchs, semble indispensable. C'est dans un tel cadre que se situe cette thèse. Nous nous appliquons notamment à contribuer au développement de méthodes d'extraction d'informations avec une contrainte de robustesse pour pouvoir, à terme, manipuler des données bruitées comme celles issues des transcriptions des vidéos de matchs de football.

Il est important de noter que ces textes issus de documents multimédias ont des caractéristiques particulières. Les textes incrustés sont connus pour fournir des informations intéressantes dans une fenêtre temporelle (par exemple, le score final dans un rapport de football [Snoek and Worring, 2003]) ; mais ces données textuelles dans les vidéos ne portent pas toutes les connaissances sur leur contenu. L'extraction d'informations à partir de la partie audio des documents multimédia est également utile. Elle se fait avec l'aide de systèmes de reconnaissance automatique de la parole ; ces systèmes montrent de bons résultats dans certains cas (par exemple pour les émissions d'actualités) [Gravier et al., 2011]. L'utilisation de ressources textuelles externes est une autre option, mais elle est limitée à certains domaines, tels que ceux du sport ou de la diffusion scientifique pour lesquels des données textuelles complétant ou commentant les vidéos sont disponibles [Buitelaar et al., 2008b].

Pour autant, ces textes ont la particularité d'être généralement de petite taille et contenant du bruit (par exemple, des erreurs de transcriptions). Ces caractéristiques les rendent difficiles à traiter. Bien sûr, la qualité et la quantité du bruit dépendent du genre vidéo. Par exemple, les transcriptions d'émissions d'actualités (journaux TV) sont moins bruitées que celles des matchs de football ; ces dernières ont en effet tendance à contenir des phrases informelles agrammaticales, incomplètes et contenant beaucoup de non-mots (par exemple, *oh ! Wow !*) auxquels s'ajoutent des erreurs de transcription. Les résultats prometteurs des recherches antérieures sur l'extraction d'informations à partir d'émissions d'actua-

[2]Quaero est un projet financé par OSEO, l'agence française de l'innovation, pour promouvoir la recherche dans le domaine du multimédia ; www.quaero.org .

lité [Gotoh and Renals, 2000; Guinaudeau et al., 2009] ont motivés l'orientation de notre travail, mais en ayant comme objectif à plus long terme le traitement de documents moins faciles.

En gardant cela à l'esprit, dans le cadre plus restreint de cette thèse, nous voulons faire un pas dans le sens de l'indexation multimédia ; nous cherchons donc à développer des modèles d'extraction d'information nécessitant peu ou pas de supervision, peu ou pas de connaissances spécialisées, et assez robustes pour être, par la suite, utilisés dans un contexte multimédia. Il est important de noter que le traitement direct des documents multimédias, transcrits automatiquement, est hors de la portée de cette thèse ; comme un premier pas vers cet objectif ambitieux, nos modèles sont testés sur des données partageant certaines similarités de complexité. Cela nous permet une comparaison plus faciles avec les approches existantes et une compréhension plus accessible des problèmes causés par cette complexité.

1.2 Travaux connexes

À ses débuts, l'extraction d'informations (en anglais *Information Extraction*, abrégé en IE dans la suite de ce document) a principalement eu pour but de détecter et reconnaître les noms des personnes et des organisations dans des documents textuels comme les journaux [Cullingford, 1978; Jacobs and Rau, 1990; Andersen et al., 1992] ou les formulaires bancaires [Lytinen and Gershman, 1986]. Il a ensuite été étendu à d'autres documents plus complexes tels que les pages Web, les messageries instantanées ou le *micro-blogging* [Verma et al., 2011]. En raison des résultats prometteurs de l'IE dans les documents textuels et la quantité croissante de données multimédias en ligne telles que l'audio, l'image et la vidéo, l'IE est maintenant utilisée de plus en plus pour manipuler les documents multimédia.

Par ailleurs, des recherches antérieures montrent l'importance des annotations pour la recherche d'images [Russell et al., 2008; Wang et al., 2006]. L'annotation multimédia est également importante, car elle facilite la phase de recherche intelligente de documents [Stamou et al., 2006]. En effet, outre l'amélioration de la gestion des documents multimédia, l'apport d'informations pertinentes contribue également à enrichir les documents multimédias. Un tel enrichissement a déjà été fait dans certaines applications telles que les émissions d'actualités [Gravier et al., 2011]. Pour chaque reportage, sur la base des informations extraites, toutes les données connexes provenant de sources différentes sont visibles en même temps à l'utilisateur. Cette information supplémentaire permet une meilleure compréhension du reportage consulté. Un autre cas possible est celui des vidéos éducatives liées à d'autres sources externes et connexes de connaissances (par

exemple, Khan Academy vidéos ou Wikipedia).

Ce type d'applications a une longue histoire et certains chercheurs ont déjà essayé d'extraire des informations à partir de tels documents pour améliorer l'indexation des documents et l'extraction. Ici, nous passons en revue les recherches qui ont quelques similitudes avec les nôtres, soit dans leurs objectifs, soit dans le type d'informations extraites.

Reidsma et al. [2003] a proposé un système appelé MUMIS pour améliorer les résultats de recherche d'informations dans des archives multimédias en utilisant un système d'extraction de l'information. Différentes sources textuelles ont été utilisées dans ce projet, y compris des articles de journaux et des transcriptions vidéo. Le manque d'information temporelle dans les journaux papier rend difficiles une utilisation directe pour détecter des événements dans des vidéos. Au contraire, la transcription automatique de la parole est marquée temporellement, mais contient plus d'erreurs à cause du bruit ajouté par la phase de transcription [Sturm et al., 2003]. Ces derniers, travaillant dans le domaine du football, ont utilisé des rapports minute-par-minute décrivant les événements les plus importants de la rencontre. Ils ont supposé que certaines informations étaient déjà fournies, telles que les noms de joueur, des équipes et de l'arbitre, avec des informations temporelles sur les scores, les cartons rouges ou jaunes et les acteurs impliqués. Notre travail est similaire au projet MUMIS en ce qui concerne les données utilisées pour l'évaluation. En effet, dans les deux cas les données sont bruitées, peuvent contenir des phrases agrammaticales (sauf pour les données de journaux qui nous n'utilisons pas). Mais dans notre recherche, les techniques développées se veulent indépendantes du domaine. En particulier, nous n'utilisons pas de liste des noms propres (noms des joueurs, par exemple dans les documents de football) pour extraire des informations connexes.

Buitelaar et al. [2008a] a construit un système appelé SOBA d'extraction d'information s'appuyant principalement sur une ontologie, avec la possibilité d'intégrer des données provenant de sources différentes. Une partie du modèle proposé, d'objectif similaire à ce que nous proposons dans cette thèse (chapitres 3 et 4), s'emploie à extraire des informations à partir de données textuelles. Son fonctionnement est le suivant. Tout d'abord, une base de règles faites à la main (liste des personnes, noms de pays et d'organisations) est utilisée pour extraire des informations préliminaires. Ensuite, un système d'IE, basé sur une analyse linguistique minimale, est utilisé pour extraire des événements simples et les noms d'entités à partir du texte. Étant donné que certains événements sont définis par des phrases complexes, une analyse syntaxique des phrases est également employée. En outre, une étape d'analyse du discours est ajoutée pour extraire les événements qui sont distribués sur plusieurs phrases. Enfin, les informations recueillies par chaque partie du système sont intégrées dans une ontologie de manière à avoir une plus grande précision. Là encore, la somme de connaissances

a priori utilisées distingue ce système de notre démarche. En outre, leurs données peu bruitées leur permet l'utilisation d'analyse linguistique fine, ce qui est impossible dans le cas de documents très bruités.

1.3 Contenu du manuscrit

QuaeroDans cette thèse, nous construisons différents modules d'extraction d'informations. Comme nous l'avons dit, nous supposons que l'information textuelle relative aux documents multimédias est bruitée, ainsi nos modèles doivent être en mesure d'y faire face. En plus de la robustesse des modèles proposés, nous cherchons également à comprendre comment extraire des informations quand il y a très peu de connaissances sur la nature des données à extraire. De ce point de vue, notre tâche d'IE est plus proche du *Data Mining* où le but est de découvrir des modèles dans les données sans supervision. Pour atteindre ces objectifs (robustesse et connaissance minimale sur les données), différentes tâches sont abordées dans cette thèse pour extraire les entités et leurs relations à partir des données textuelles.

Premièrement, nous proposons un nouveau modèle pour l'extraction supervisée de relations entre entités dans le chapitre 2. Comme les données textuelles annotées de documents multimédia n'étaient pas disponibles au début de la thèse, nous avons testé le modèle d'extraction de relations dans des textes biomédicaux du challenge [Nédellec, 2005]. Nous avons choisi ce jeu de données parce qu'il a des propriétés similaires (en termes de complexité des phrases et la difficulté de détection des entités) aux données finales visées, et parce qu'il nous permet de nous comparer à l'état de l'art. Ensuite, afin d'avoir des expériences sur des données plus étroitement liées à notre objectif final, nous avons décidé de travailler sur des données de football. Pour chaque match de football, outre la vidéo, un rapport textuel est recueilli à partir du Web. Dans ces rapports, les événements importants pour chaque minute du match sont décrits. Ensuite, les textes recueillies, à partir des vidéos (transcriptions) et des rapports textuels ont été annotées [Fort and Claveau, 2012]. La seconde tâche de cette thèse porte sur la découverte et le regroupement des relations entre noms propres dans le texte (voir le chapitre 3). Contrairement à la première tâche, qui a besoin de données d'entraînement annotés par un expert, le modèle proposé ne nécessite pas de supervision. La troisième tâche est définie comme un problème de découverte et de *clustering* sémantique d'entités (voir le chapitre 4). Enfin, quelques remarques finales et des perspectives pour les travaux futurs sont présentées dans le chapitre 5.

Chapitre 2

Extraction supervisée de relations

Depuis les années 90, beaucoup de travaux de recherche ont été consacrés au problème de l'acquisition de connaissances sur corpus, que ce soit pour des terminologies, des cas particuliers de vocabulaire (par exemple, les entités nommées), ou des relations entre les mots. Le travail présenté dans ce chapitre se concentre sur ce dernier type d'acquisition, c'est-à-dire l'extraction les relations. Dans ce chapitre, nous proposons un modèle supervisé pour extraire et classer les relations.

Afin d'évaluer le modèle proposé, nous expérimentons sur des données réelles possédant des propriétés communes à un texte transcrit. Nous supposons que les textes transcrits sont bruités, donc on ne peut pas utiliser des techniques d'analyse linguistique profonde, trop sensibles à la qualité des données. En outre, nous cherchons à travailler avec des données relativement petites similairement à ce que l'on peut obtenir pour une émission vidéo. Nous évaluons donc nos modèles sur une tâche réputée difficile du domaine bio-médical : la détection d'interaction protéine-protéine (PPI).

Le but de cette tâche est de trouver des paires de protéines dans les phrases telles qu'une protéine est décrite comme régulant, bloquant ou se liant à une autre. En génomique fonctionnelle, ces interactions, qui ne sont pas disponibles en base de données structurée, mais dispersés dans des revues scientifiques, sont essentielles pour déterminer la fonction des gènes. Afin d'extraire les PPI, les textes (articles scientifiques) qui contiennent les interactions doivent être analysés. Deux types d'analyse linguistique peuvent être utilisés à cette fin : profonde ou de surface, qui sont expliquées dans le chapitre correspondant de l'annexe.

Dans ce chapitre, nous défendons l'utilisation d'attributs linguistiques simples pour ces tâches d'extraction de relations. D'abord, nous montrons que ces éléments fiables et faciles à obtenir peuvent être efficacement utilisés comme attributs pour l'apprentissage automatique, ce qui permet d'entraîner de bons

systèmes d'extraction de PPI. Pour promouvoir cette idée, nous proposons donc un système simple mais original, appelée LM-kNN basée sur la modélisation linguistique, qui surpasse les systèmes de l'état-de-l'art.

Dans l'extraction PPI, l'objectif est de prédire s'il ya une l'interaction entre les deux protéines. Dans un tel cas, la relation est dirigée, c'est-à-dire avec une entité agent et une autre dite cible. Par exemple, dans la phrase rapportée dans la figure 2.1, les entités (protéines) sont en gras, et il existe une relation entre *GerE* et *CotD* pour laquelle *GerE* est l'agent et *CotD* est la cible.

GerE *stimulates* ***cotD*** *transcription and inhibits* ***cotA*** *transcription in vitro by* ***sigma K*** *RNA polymersase, as expected from in vivo studies, and, unexpectedly, profoundly inhibits in vitro transcription of the gene (****sigK****) that encode* ***sigma K****.*

FIG. 2.1: Sample sentence for protein-protein interaction

L'approche que nous proposons tient compte de l'aspect séquentiel de la tâche à l'aide de modèles de langage n-gramme. Ainsi, une relation est représentée par la séquence des lemmes survenant entre l'agent et la cible, si l'agent apparaît avant la cible, ou entre la cible et l'agent dans le cas contraire. Un modèle de langage est construit pour chaque exemple Ex, c'est-à-dire que l'on collecte les probabilités d'apparition des n-grammes, estimées sur la base de leurs occurrences dans Ex. Ce modèle de langage est noté $\mathcal{M}_{Ex}$. La classe (LTR, RTL ou aucune) de chaque exemple est également mémorisée.

Compte tenu d'une relation candidate (c'est-à-dire deux protéines ou gènes dans une phrase), il est possible d'évaluer sa proximité avec n'importe quel exemple, ou plus précisément la probabilité que cet exemple ait généré le candidat. Notons $C = \{w_1, w_2, ..., w_m\}$ la séquence de lemmes entre deux protéines. Pour des n-grammes de n lemmes, cette probabilité est classiquement calculée comme suit :

$$P(C|\mathcal{M}_{Ex}) = \prod_{i=1}^{m} P(w_i|w_{en}..w_{i-1}, \mathcal{M}_{Ex})$$

Comme pour tout modèle de langage dans la pratique, les probabilités sont lissées afin d'empêcher que les n-grammes non vus dans Ex donne un score de 0 à l'ensemble de la séquence. Dans les expériences rapportées dans l'annexe correspondante, nous considérons des bigrammes de lemmes, interpolés avec l'ordre inférieur (unigramme) et associés à un lissage *absolute discounting* [Ney et al., 1994].

Afin d'éviter que des exemples avec de longues séquences soient privilégiées, la probabilité de générer l'exemple à partir du candidat ($P(Ex|\mathcal{M}_C)$) est également prise en compte. Finalement, la mesure de similarité entre un exemple et un candidat est donc :

$$Sim(Ex, C) = \min\left(P(Ex|\mathcal{M}_C), P(C|\mathcal{M}_{Ex})\right).$$

La classe est finalement attribuée au candidat par un algorithme de k-plus proches voisins : les 10 exemples les plus similaires sont trouvés et un vote à la majorité est effectué. Cette technique d'apprentissage paresseux est censé être plus adaptée à ce type de tâches que celles basées sur des modèles proposées dans la littérature, car il prend mieux en compte la diversité des façons d'exprimer une relation (voir l'annexe correspondante pour une discussion sur cette question).

Chapitre 3

Découverte de relations

Dans le chapitre précédent, nous avons proposé un modèle supervisé pour l'extraction de relation en la modélisant comme un problème de classification. Nous avons montré l'efficacité du modèle de langue dans un tel contexte. Ces types de modèles supervisés sont utiles pour les problèmes sur lesquels nous avons des connaissances a priori sur les relations, et notamment quand les types de relations sont pré-définis. Mais dans certains cas, nous ne souhaitons ou ne pouvons faire aucune hypothèse sur les relations qui seraient intéressantes. Dans ce chapitre, nous nous concentrons donc sur les approches non supervisées pour découvrir les relations entre les entités. Selon les expériences dans le chapitre précédent, nous supposons là encore que l'analyse linguistique de surface peut être efficace pour l'apprentissage non supervisé.

Considérant la découverte de relations comme un problème d'apprentissage non supervisé, c'est à dire un *clustering*, il n'est pas nécessaire d'avoir de connaissances a priori sur les données. Mais il est nécessaire de savoir quelles caractéristiques des objets sont importantes pour détecter et grouper sémantiquement les relations.

Dans notre problème de découverte Relation, l'objectif est de découvrir les relations entre les noms propres dans une phrase. Nous modélisons donc le problème comme une tâche de *clustering* où l'objectif est d'attribuer un ensemble de relations au même groupe tel que deux relations au sein du même groupe ont plus de similitudes que deux relations appartenant à deux clusters différents. Tout d'abord, nous devons définir la mesure de similarité entre les relations en fonction de leurs caractéristiques. Nous considérons les informations contextuelles comme attributs des relations, de sorte que chaque relation est représentée comme un vecteur basé sur des informations contextuelles (voir l'annexe correspondante).

Dans notre solution pour découvrir les relations entre noms propres, nous recueillons tout d'abord toutes les relations potentielles du texte en collectant chaque occurrence de paires de noms propres. Il a déjà été démontré que la

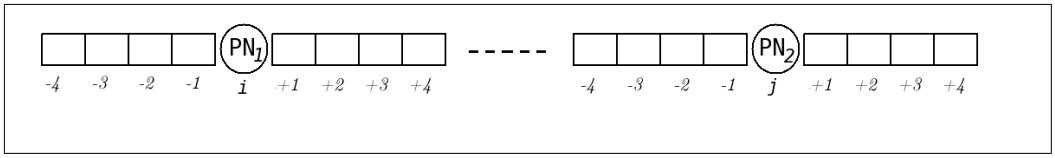

FIG. 3.1: Paire d'entités et leur information contextuelle

suppression de données bruitées peut améliorer l'extraction d'information non supervisée [Wang et al., 2011]. Nous adoptons aussi ce précepte puisque nous utilisons des filtres pour éliminer les données non-pertinentes (paires d'entités sans aucun rapport avec les relations potentielles). Par exemple, la distance (nombre de mots) entre les paires d'entités est une propriété importante pour une relation. Nous supposons que s'il y a plus d'un nombre fixé de mots entre la paire de noms propres, nous pouvons supprimer la relation des candidats. Afin d'étudier l'importance de cette contrainte et optimiser ce filtre, nous examinons différents seuils sur les données et analysons les résultats (voir l'annexe correspondante).

Le cœur de notre système consiste à considérer les relations et leurs similitudes dans un graphe tel que chaque nœud du graphe est une relation et chaque arc est la similarité entre les relations concernées. Nous commençons par construire la matrice de similarité A dans laquelle chaque cellule a_{ij} est la similarité entre les relations r_i et r_j. Ensuite, nous utilisons l'algorithme de clustering de Markov regrouper les noeuds similaires du graphe. La façon de calculer la matrice de similarité est la principale contribution de ce chapitre.

Différentes sources d'information textuelles sont généralement utilisées pour la découverte de relations, telles que le chemin entre les entités d'un arbre syntaxique [Zhang et al., 2005; Shinyama and Sekine, 2006] ou la séquence de mots ou des étiquettes autour des entités dans la phrase [Hasegawa et al., 2004; Wang et al., 2011], ou encore une combinaison de ces attributs [Bollegala et al., 2010]. Dans notre cas, pour représenter une relation, nous utilisons la séquence de mots (n-gramme) avant et après la première et la seconde entité comme le montre la figure 3.1. Par exemple, dans la phrase "*Fandel sanctionne Jurietti Pour Un tacle trop appuyé sur Totti*", il y a trois noms propres et trois relations possibles entre eux (sans direction). Les deux 3-grammes autour des paires de noms propres de chaque relation sont énumérés dans le tableau 3.1.

Comme tous les n-grammes n'ont pas la même importance pour la relation, nous calculons leur poids en nous appuyant sur un tf-idf. Pour cela, nous considérons les phrases comme des documents et le texte entier comme le corpus dans la définition du tf-idf.

Afin de diminuer l'éparsité des données, tous les noms propres sont remplacés par des étiquettes de la classe sémantique (nom des joueurs, arbitres, etc...). Dans les expériences rapportées dans ce chapitre, cette liste de noms et classes

Relation	Information contextuelle (n-gram)
Fandel-Jurietti	[sanctionne] , [pour un tacle] , [un tacle trop]
Fandel-Totti	[sanctionne Jurietti pour] , [tacle trop appuyé] , [trop appuyé sur] , [.]
Jurietti-Totti	[Fandel sanctionne] , [pour un tacle] , [un tacle trop] , [tacle trop appuyé] , [trop appuyé sur] , [.]

TAB. 3.1: Deux 3-grammes autour de chaque paire de noms propres

est construite manuellement, mais il est également possible de construire automatiquement la liste à l'aide des techniques que nous proposons dans le chapitre suivant. En outre, nous utilisons les lemmes des mots plutôt que des mots, ce qui diminuent la rareté des données.

Dans la tâche du chapitre précédent, nous avons examiné certaines fonctions de similarité classiques telles que cosinus et Jaccard. Nous avons déjà montré que la fonction de similarité basée sur les modèles de langue peut être plus performants que les fonctions de similarité classiques. Dans ce chapitre, nous nous en inspirons pour définir une nouvelle mesure de similarité. Celle-ci est basée sur la probabilité de génération d'une relation par rapport aux autres. Considérant deux relations R_1 et R_2, nous définissons la similarité entre ces deux relations comme le minimum des probabilités moyennes d'avoir R_1 si R_2 est donnée et vice-versa ; voir l'équation ci dessous.

$$SIM(R_1, R_2) = \min\{F(R_1|R_2), F(R_2|R_1)\} \tag{3.1}$$

où $F(R_1|R_2)$ est la probabilité moyenne de R_1 sachant R_2, calculée sur la base de l'équation 3.2. Là encore, l'utilisation du minimum nous permet d'éviter de favoriser les relations avec de longues séquences de mots.

$$F(R_1|R_2) = \frac{1}{n}\sum_{i=1}^{n} P(NG = ng_{1i}|R = R_2) \tag{3.2}$$

La probabilité conditionnelle de ng_{1i} sur la base des données fournies de R_2 peut être facilement calculée par une estimation de vraisemblance maximale définie dans l'équation ci-dessous, où chaque n-gramme est une séquence de mots. Par exemple, avec un 3-grammes, chaque n-gramme est défini comme $ng = w_1w_2w_3$ et $c(w_1, w_2, w_3, R_2)$ est le nombre d'occurrences de la séquence $w_1w_2w_3$ dans R_2.

$$P(ng|R2) = \frac{c(w_1, w_2, w_3, R_2)}{c(w_1, w_2, R_2)} \tag{3.3}$$

Comme nous l'avons déjà évoqué, il est nécessaire d'utiliser une technique de lissage pour éviter les probabilités nulles. Parmi toutes les méthodes de lissage, nous avons utilisé de nouveau l'*absolute discounting*, car il a conduit à de bons résultats dans le problème d'extraction de relations. De plus, il est simple à mettre en œuvre et ne nécessite pas de données d'entraînement. Cependant, la technique de lissage utilisée dans cette approche a tendance á entraîner une similarité minimale entre toutes les paires de relations ; cela rend difficile l'obtention de grandes différences entre les relations. Pour résoudre ce problème, nous proposons une fonction de similarité plus discriminante. Cette fonction élève simplement la similarité à une puissance p :

$$SIM(R_1, R_2) = (\min\{F(R_1|R_2), F(R_2|R_1)\})^p \tag{3.4}$$

Chapitre 4

Classification non-supervisée d'entités

Dans les deux chapitres précédents, nous avons proposé des modèles pour l'extraction supervisée et la découverte de relations. Pour les besoins de ces travaux, nous avons supposé que les entités étaient identifiées dans le texte, mais la détection et la classification des entités (ou clustering) est une autre tâche difficile mais essentielle pour l'extraction d'information. Dans ce chapitre, nous nous intéressons à ce problème, et plus précisément à la classification des entités extraites de textes — dans notre cas des noms propres — en fonction de leurs contextes dans un corpus. Il est intéressant de noter que cette tâche est proche de la reconnaissance d'entités nommées (NER), mais en diffère à certains égards. En effet, l'objectif de la reconnaissance d'entités nommées est de localiser et de classer les entités nommées dans des groupes prédéfinis, tels que les noms de personne, lieu, organisation... La localisation et la classification peuvent se faire soit en une seule étape, soit en deux étapes consécutives, mais pour ces deux sous-tâches, la plupart des systèmes NER reposent sur des modèles supervisés, s'appuyant sur des données annotées manuellement. Pourtant, dans ce travail, notre objectif est légèrement différent de cette stricte définition puisque nous cherchons à construire des classes d'entités sans aucune supervision ou présupposition sur les classes. Plus précisément, nous voulons grouper les noms propres (PN) en fonction de leurs similitudes, sans une connaissance a priori sur les classes possibles.

Le choix de la fonction de similarité est fortement tributaire de la représentation utilisée pour décrire les entités. Dans ce chapitre, nous étudions l'utilisation d'une nouvelle représentation qui devrait surpasser celles couramment utilisées. En effet, la méthode classique de calcul des similarités cherche à construire un vecteur caractéristique, ou sac-de-mots, pour chaque entité, puis á utiliser les fonctions de similarité classiques comme le cosinus. Dans la pratique, les caractéristiques sont contextuelles, telles que les mots ou les n-grammes au-

Phrase
Zigic donne quelques frayeurs à Gallas et consorts en contrôlant un ballon chaud à gauche des 16 mètres au devant du Gunner.

PN	**n-gramme**
Zigic	donne quelques frayeurs — quelques frayeurs à
Gallas	donne quelques frayeurs — quelques frayeurs à, et consorts en — consorts en contrôlant
Gunner	mètres au devant — au devant du

Table 4.1: N-gramme collecté pour un nom propre avec une fenêtre de 4 mots

tour des différentes occurrences de chaque entité. Ici, nous proposons d'utiliser une représentation alternative pour les entités, appelé sac-de-vecteurs, ou sac-de-sacs-de-mots, en s'inspirant de certains travaux en recherche d'images [Gosselin et al., 2007]. Dans ce nouveau modèle, chaque entité n'est pas définie par un vecteur unique, mais par un ensemble de vecteurs, dans lequel chaque vecteur est construit sur la base des caractéristiques contextuelles (mots ou n-grammes environnant) d'une seule occurrence de l'entité dans le corpus. Les fonctions usuelles de similarité ou de distance, y compris le cosinus, le Jaccard et les distances euclidiennes peuvent être facilement étendues pour gérer cette nouvelle représentation. Dans ce chapitre, ces systèmes de représentation et distances sont évalués sur notre tâche de clustering de noms propres.

Différentes caractéristiques contextuelles ont été explorées dans nos expériences, basées sur les mots, lemmes ou n-grammes qui entourent chaque occurrence d'un PN. Dans les expériences rapportées dans l'annexe correspondant à ce chapitre, nous présentons uniquement les résultats pour les caractéristiques qui ont donné les meilleurs résultats. Ils sont basés sur des 3-grammes recueillis dans une fenêtre de 4 mots avant et après chaque occurrence de PN dans une phrase. Un exemple de collecte des n-grammes est donné dans le tableau 4.1.

Différents systèmes de pondération pour les n-grammes recueillis ont également été étudiés (cf. annexe), afin de donner moins d'importance aux n-grammes très communs. Là encore, à des fins de simplicité, nous ne présentons que ceux qui donnent les meilleurs résultats, à savoir une version modifiée du tf-idf [Momtazi et al., 2010] dans laquelle le score idf est calculé en considérant les phrases comme des documents (ces phrases étant courtes, le tf est presque toujours égal à 1, le système de pondération est donc surtout un pur idf).

Dans le modèle standard de représentation sac-de-mots (BoW), pour chaque PN détecté dans le corpus, un vecteur de caractéristiques unique pondéré est

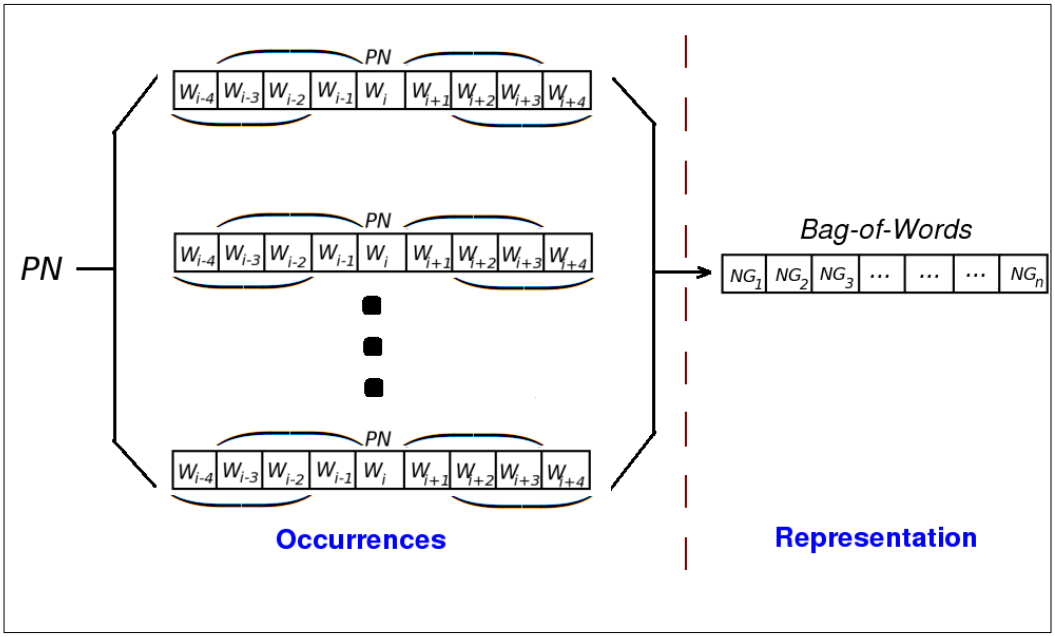

Figure 4.1: Sac-de-mots (n-grammes) pour représenter un PN

construit sur la base des n-grammes avant et après toutes les occurrences du PN dans l'ensemble du corpus. Grâce à son éparsité, le vecteur résultant permet de calculer des distances très efficacement. Cependant, dans une telle représentation, les n-grammes en provenance des différentes occurrences d'un PN sont mélangés (voir figure 4.1). Ainsi, sur la base de cette représentation, la comparaison des deux entités ne peut être faite au niveau de l'occurrence mais en fonction d'un profil global. La représentation sac-de-vecteurs que nous proposons d'utiliser tente de maintenir les bonnes propriétés de la représentation vectorielle tout en offrant une représentation plus souple, gardant distinctes les occurrences.

Dans notre modèle, chaque PN dans le texte est représenté par un sac de vecteurs dans lequel chaque vecteur est un sac-de-mots standard pour chaque occurrence du PN. Prenons un PN P_1 ; sa représentation BOV est définie dans l'équation 4.1.

$$BOV(P_1) = \{b_{11}, b_{12} \dots b_{1i} \dots b_{1r}\} \tag{4.1}$$

lorsque r est le nombre d'occurrences de P_1 dans le corpus et b_{1i} est un vecteur représentant la ième apparition de P_1 (comme un arc) dans le corpus.

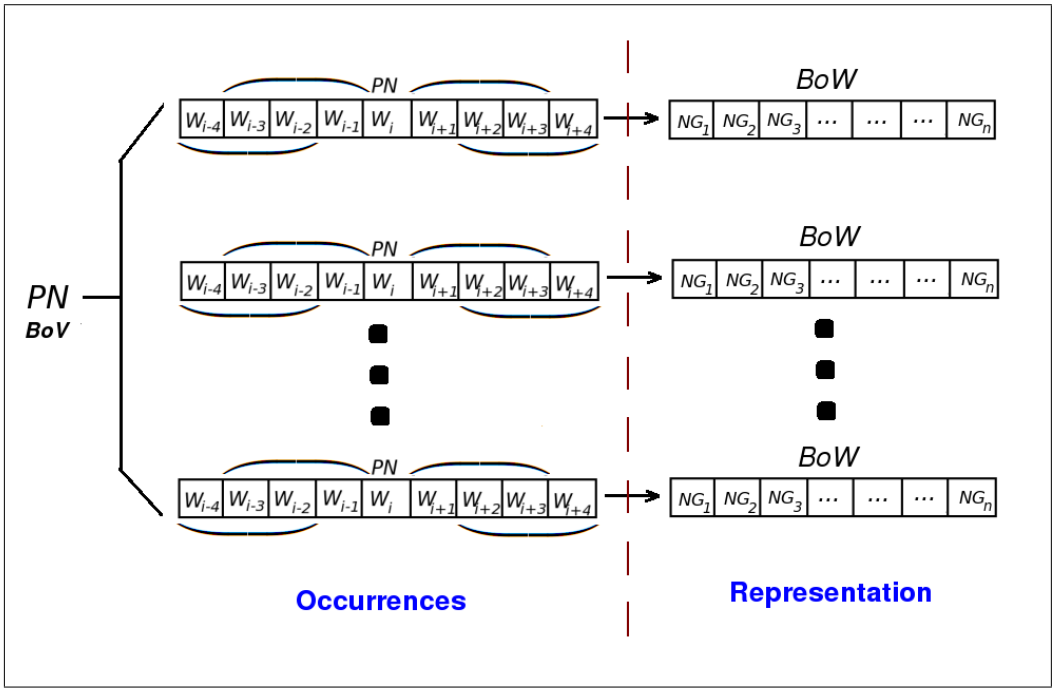

Figure 4.2: Sac-de-Vecteurs n-grammes pour PN

Chapter 5

Conclusion et perspectives

Dans ce dernier chapitre, nous présentons quelques réflexions globales sur les limites de nos approches et présentons quelques perspectives. En effet, nous ne prétendons pas avoir résolu tous les problèmes dans ce large domaine de recherche en 3 ans. Dans l'annexe correspondante, nous passons en revue les limites de chacun des modèles proposés et donnons quelques éléments de solutions possibles. Ces limites concernent pour la plupart une complexité algorithmique des approches pouvant être préjudiciable en cas de traitement de très grand jeux de données. Cependant, comme nous l'avons expliqué, notre but premier était plutôt d'avoir un système robuste plutôt que scalable.

Enfin, nous proposons ci-après quelques sujets de possibles recherches futures dans le cadre de l'extraction d'information à partir de documents multimédia.

Approches multimodales. Les approches multimodales peuvent améliorer les performances des modèles en intégrant des informations extraites de différentes sources en un seul système. Comme nous l'avons mentionné précédemment, différentes sources de données textuelles peuvent en effet être trouvées pour les documents multimédia (par exemple, les textes externes comme les résumés minute-par-minute des matchs de football). La redondance des résultats provenant de différentes sources peut aider à valider chacune des données extraites ou à y détecter des contradictions. Par exemple, dans les résumés de football, si le score du match extrait dans le texte transcrit est différent de celui extrait des journaux, on peut supprimer les informations de la première source en supposant que l'extraction d'informations à partir du texte transcrit contient vraisemblablement plus d'erreurs par rapport aux journaux écrits. En outre, les informations provenant de différentes sources peuvent être complémentaires et, par conséquent, les intégrer aide à obtenir un plus grand nombre d'informations pour chaque période de temps dans le document. Mais le défi le plus important dans cette approche est la définition effective de ce

modèle d'intégration permettant de gérer la fiabilité et la complémentarité des données. Cela est en soi un vaste mais intéressant sujet de recherche .

Amélioration automatique du texte transcrit. Cette piste est sans doute la plus importante pour enfin placer complètement l'IE dans le domaine des documents multimédias. En effet, l'une des observations au début de notre travail de recherche a été les très mauvais résultats de la reconnaissance automatique de la parole (RAP) sur la vidéo tout-venant, et plus particulièrement sur les vidéos de football. Le principal problème est causé par le niveau élevé de bruit dans le stade, qui existe toujours dans ce genre de vidéos. Différentes techniques ont été proposées dans la littérature pour rendre un système ASR robuste contre un tel niveau de bruits (par exemple, [Al-Haddad et al., 2009]). Elles semblent indispensables pour obtenir une matière première suffisante pour les traitements ultérieurs. En outre, il convient de mentionner que la classe de mots la plus fréquente dans les reportages vidéo de football est celle des noms propres. Or, la plupart de ces noms propres sont inconnus du système de RAP, ce qui diminue inévitablement et fortement ses performances. Pour répondre à ce problème, Lecorvé et al. [2008] ont proposé des techniques pour adapter le modèle de langage du système de RAP pour un domaine spécifique. En outre, une liste de noms d'entités (par exemple, les noms propres) peut être utilisée pour une telle adaptation. Or, notre modèle dans le chapitre 4 permet justement de détecter ces classes d'entités susceptibles d'être inconnues et ainsi de construire des listes permettant ensuite d'améliorer les performances des systèmes de RAP.

Part II

Toward robust Information Extraction for multimedia documents

Chapter 6

Introduction

During the last decade, huge amounts of multimedia documents have been generated (for example, 72 hours of video are uploaded to YouTube each minute). It is therefore important to find a way to manage this data. Every approach to facilitate this process requires to have a deep understanding of the content of the documents. There are two main approaches to get such insights into the multimedia documents, either by extracting information from the document or by using related data from external sources (such as the Web). In the first approach, a variety of information can be extracted from the documents such as sequences of video shots [Sivic and Zisserman, 2003], textual information [Jung, 2004; Elagouni et al., 2012] and audio content[Lawto et al., 2011] (see [Hu et al., 2011] for more information). In the second approach, some researchers have proposed models using external textual resources, and particularly textual ones, to help the management of multimedia documents [Buitelaar et al., 2008b; Reidsma et al., 2003].

Objectives

The work presented in this thesis belongs to this second approach. It originates from a broader framework concerned by the indexing of multimedia document, as it has been conducted in the INRIA/TexMex project[1] and has been funded by the Quaero project[2]. In this multimedia indexing context, any linguistic or textual information related to the processed document is a very valuable input to describe its content. In videos for instance, it can be overlaid texts, related Web sites, or more commonly, speech transcripts.

As an example, assume a user looking for precise answer to the following

[1] www.irisa.fr/texmex

[2] This thesis was achieved as part of the Quaero Programme, funded by OSEO, French State agency for innovation whose goal is to promote research for automatic analysis and classification of multimedia documents.

information needs:

- All goal scenes of the match between Germany and Italy in Euro2012.
- All foul scenes resulting to yellow card in Euro2012.

To respond to these needs, an annotation or semantic information extraction of the football report is needed to find the appropriate period of time in the corresponding unstructured data. So, this thesis is defined as an Information Extraction task for texts, such as those extracted or related to videos of matches.

However such textual data have special characteristics. Overlaid texts are known to deliver interesting pieces of information in a period of time (for example, the final score in a football report [Snoek and Worring, 2003]); but these textual data within the videos do not carry all knowledge about the videos. Extracting information from the audio part of multimedia documents is also helpful with the help of Automatic Speech Recognition (ASR) systems that show good results (e.g. for news broadcasting) [Gravier et al., 2011]. Using external resources is another option; but it is limited to certain domains (e.g. sport reports or scientific presentations) where textual data are available [Buitelaar et al., 2008b].

Yet, such texts have the peculiarity of being usually small and noisy. These characteristics make them challenging to process. Of course, the quality and the amount of the noise depend on the video genre. For example, transcriptions of broadcasting news are less noisy than those of football reports, as the latter may contain non grammatical sentences, lots of non-words (e.g. oh! Wow!), incomplete or informal sentences. Promising results of prior researches about information extraction from broadcasting news [Gotoh and Renals, 2000; Guinaudeau et al., 2009] motivated us to work on transcribed texts; but for documents with informal sentences (e.g. football match reports) which is a more challenging task.

Keeping that in mind, in this thesis, we want to take a step in the direction of text-based multimedia indexing and thus we aim at developing Information Extraction models requiring small supervision or expert knowledge and robust enough to be, later, used in a multimedia context. It is important to note that processing directly multimedia documents or automatically transcribed texts is out of the scope of this thesis; as a first move towards this ambitious objective, our models are tested on data sharing some of these challenging characteristics, but also allowing comparison with existing approaches and in-depth understanding of the problems that they cause.

Related Work

At its early ages, Information Extraction (IE) dealt with extracting information such as people names and organizations from text document like newspapers [Cullingford, 1978; Jacobs and Rau, 1990; Andersen et al., 1992] or bank forms

[Lytinen and Gershman, 1986]; it was then expanded to more complex text documents such as web pages, micro-blogging posts [Verma et al., 2011]. Due to the promising results of IE in textual documents and the increasing amount of online multimedia data such as audio, image and video, IE is now used to build structured multimedia documents too. These structured documents are more easy to manage compared to the originals.

Furthermore, previous researches on image annotation show the importance of annotation for image search and retrieval [Russell et al., 2008; Wang et al., 2006]. Multimedia annotation is also important, because it facilitates search, retrieval and intelligent processing of documents [Stamou et al., 2006]. Namely, in addition to improve the management of multimedia documents, it also helps to enrich the multimedia documents in order to have more effective presentation. This enrichment has already been done in some applications such as news broadcasting [Gravier et al., 2011]. For each report, based on the extracted information, all related data from different sources can be shown at the same time to a user. This extra information helps audience to have smoother understanding. For example, imagine the case of having educational videos linked to other external related sources of knowledge (e.g. Khan Academy videos or Wikipedia). It has a long history and some researchers have already tried to extract information from multimedia documents to improve documents indexing and retrieving. Here, we review the researches which have some similarities either in their goals or in the type of extracted information.

Reidsma et al. [2003] introduced a system called MUMIS to improve the results of Information Retrieval in multimedia archives by using an Information Extraction system. Different sources of textual information were used in this project including newspaper reports and video transcription. The lack of temporal information in newspaper reports make them difficult to be used directly for event detection in the videos. On the contrary, automatic speech transcription contains temporal information but more errors because of the noise [Sturm et al., 2003]. Additionally, they used a minute-by-minute report containing most important events of the match. They assumed that some information has already been provided such as player, team and referee names with temporal information about scores, red or yellow cards and players who are involved. Our research is similar to the MUMIS project based on their data used for evaluation. In both researches, the data is noisy and have ungrammatical sentences (except for the newspaper data which we do not use). But our research is about Information Extraction from multimedia documents independent of the domain. Moreover, we do not use any list of proper nouns (e.g. player names in football documents) to extract related information.

Buitelaar et al. [2008a] built a system called SOBA which is an ontology-based Information Extraction system with the possibility of integrating data

from different sources. Part of the proposed model, which is similar to what we propose in this thesis (in chapters 9 and 10), is a way of extracting information from textual data. First of all, a hand made rule-based model with a gazetteer component (list of persons, countries and organizations names) is used to extract some preliminary information. Different information including player names and the final score are collected in this phase. Then, an IE system, based on minimum linguistic analysis, is defined to extract simple events and entity names from the text. Since some events are defined via complex sentences, a syntactic analysis of sentences is also used. Additionally, a discourse analysis is defined to extract events that are distributed in more than one sentence. Finally, the information collected by each part of the system is integrated in one knowledge-based system in a way to have higher precision. This research is also different from ours since they used a list of person names as a main component of the system. Moreover, they used some less noisy textual data to improve the overall system performance; but we aim to work only on noisy data which prevent us to utilize syntactic or semantic analysis in order to extraction information.

The Thesis

In this thesis, we build different modules of Information Extraction. As we said, we assume that textual information related to multimedia documents is noisy, thus models for extracting information from it must be able to deal with this noise. In addition to the robustness of the proposed models, we are also interested in understanding how to extract information when there is very few assumption about the kind of the extracted data. In this way, our Information Extraction task is more similar to Data Mining where the goal is to discover patterns in the data. To meet these goals (robustness and minimum assumption about the data), different tasks are defined in this thesis to extract entities and their relations from the textual data

First, we propose a new model for extracting relations between entities in Chapter 8. Since the annotated textual data for multimedia documents was not available at the beginning of the thesis, we examined the model for extracting relations in Biomedical texts shared task [Nédellec, 2005]. We selected this task because the data had similar properties (in terms of sentence complexity and difficulty of entity detection) to the our final data set and because it allows us to compare with other existing technique. We propose a model based on the concept of Language Modeling to estimate the similarity between two relations. The final model predicts the relation label based on a voting among the nearest relations. Combination of the similarity estimation with the nearest neighbor model outperformed the state-of-the-art systems in the shared task both in the evaluation based on labeled data and those based on unseen data (see Section 8.4).

Then, in order to have experiments on data more closely related to our final

objective, we decided to work on football reports. For each football match, a textual report is collected from the Web. In these reports, important events for each minute of the match is provided. Then, the collected information including videos and textual reports was annotated [Fort and Claveau, 2012].

The second task in this thesis is defined as discovering and clustering relations between proper nouns in the text (see Chapter 9). In contrast to the first task, which needs labeled training data, the proposed model does not require supervision. One of our major contribution in this work is in the way that we realize this discovery task as a clustering one, which, in turn, relies on an effective definition of the similarity between two relations based on Language Modeling.

The third task is defined as discovering and semantic clustering proper nouns in the text (see Chapter 10). For that purpose, we propose a new data representation model to define each proper noun in the text. Each occurrence of proper nouns is considered separately when we want to find out the similarity between them. We show that this new model outperforms classical data representation when applied to our data sets arising from football reports.

To sum up, the manuscript is structured as follows. We first provide a review of the literature of information extraction followed by some related technical backgrounds including some Machine Learning and Natural Language Processing techniques which are used or needed later. Then, our three contributions (Relation Extraction, Relation Discovery and Proper Noun Clustering) are presented and discussed in the three following chapters. Finally, conclusive remarks and some perspectives for future work are presented in Chapter 11.

Chapter 7

Background

Different models and approaches are proposed in this thesis which use some common background in Machine Learning and Natural Language Processing. In this chapter, only some background needed to make the proposed models clear is explained.

First, we review different aspects of Information Extraction (IE) including applications and types of information that we can expect to extract. Then, in Section 7.2, some Machine Learning algorithm of two main groups, supervised and unsupervised are reviewed. Different Natural Language Processing techniques, including linguistic and statistical analysis are discussed in Section 7.3.

7.1 Information Extraction

Information Extraction gets a lot of attention due to the explosion of the data over the Internet. Originally, it was defined for extracting information from text. The promising results of IE on textual content have motivated researchers to apply the very same techniques to audio, image and video processing.

In this section relying on [Sarawagi, 2008], we first review some applications of Information Extraction. Then, the different types of data that can be extracted from text are explained. Finally, different approaches used in IE are briefly reviewed.

7.1.1 Applications

Extracting information from unstructured data (i.e. texts, videos, TV programs) is useful in various applications. Here, we list some of these applications in four different groups including enterprise applications, personal information management, scientific and web oriented applications [Sarawagi, 2008].

Enterprise Applications

- *Event Tracking:* One of the earliest application of information extraction is automatically *tracking news* events from news sources. For instance, the popular Message Understanding Conference [Grishman and Sundheim, 1996] was about tracking events and entities in such news reports. Once extracted, the data can be useful to enrich the original data with external information. For example, Gravier et al. [2011] proposed a model to connect related news TV reports to each other and enrich TV programs by finding related external source of knowledge based on the extracted information.

- *Customer Care:* All companies collect different forms of unstructured data from their customers. These data are a kind of treasure for the companies, if they have the possibility of converting them to structured data. Automatic identification of product names and attributes from customer emails and linking them to specific transactions in the sales database [Chakaravarthy et al., 2006] are two good examples of using IE for customer cares.

- *Data Cleaning:* Data warehouse cleaning processes aims to detect and correct inaccurate records from databases. Data segmentation can be useful to reach these goals. As an example, splitting users addresses into small parts such as number, street name and city can help this process. Splitting the address also facilitates querying the database. Often, for one user there are more than one address in the system; this problem can be solved by extracting detailed information from user records [Agichtein and Ganti, 2004].

Personal Information Management

The volume of information has not only explodes in the World Wide Web (WWW) but also in personal information. Personal Information Management (PIM) systems are designed to automatically organize personal data such as emails, documents and projects in a structured inter-linked format [Dong and Halevy, 2005]. Extracting information from unstructured (local or cloud) files is an essential task in PIM. For example, we should be able to extract automatically phone numbers and addresses from emails and add them to our address book.

Scientific Applications

Extracting information from bio-informatics has gone beyond the earlier named entities extraction. IE has been used to extract biological objects such as proteins and genes and their interactions. These entity names are more complex than classical ones such as people and organization names. The complexity of sentences in scientific papers is also different from normal texts (see Chapter 8).

Web Oriented Applications

- *Citation Database:* Different citation databases on the Web have been built based on the extracted information from conference websites or personal web pages. Citeseer [Lawrence et al., 1999] and Cora [McCallum et al., 1999] are two examples of such applications.

- *Opinion Database:* There are many websites and forums that store opinions about different topics such as products, books or even politicians in unstructured formats. These are valuable information that needs to be structured in order to be usable for analysis [Pang and Lee, 2008]. This family of studies encompasses the popular "Opinion Mining" or "Sentiment Analysis" domains. For example, it is useful to find out the users' opinions about the product features [Lee et al., 2008]. As another example, it is important for politicians to know how people react to their talk or how to evaluate their public image automatically [Sarmento et al., 2009].

- *Comparison Shopping:* Automatically extracting information from shopping websites to let the customers compare a product in different shops is another interesting application of IE [Doorenbos et al., 1997].

7.1.2 Types of Extracted Information

Although IE was originally used for text, it was later applied on other kinds of documents such as speech [Kubala et al., 1998], images and videos [Kiryakov et al., 2004]. In this thesis the focus is set on extracting information from text. More precisely, we aim at processing informal text either clean or noisy or with that may be extracted from multimedia documents ungrammatical sentences. Different types of information are recognized by IE systems including entities, relations between entities, adjectives describing entities and higher order structures such as tables and lists.

- *Entities* in text are defined mostly as noun phrases which can correspond to one token or a combination of more tokens in a sentence. The most popular form of entities is *named entities* like names of persons, locations and companies. Named Entity recognition was introduced first in MUC-6 [Grishman and Sundheim, 1996] to extract three kinds of names: person, location and organization.

- *Relations* are defined over two or more entities. For example, "is employee of" is defined as a relation between a person and an organization. In some cases, a binary relation cannot express the relation completely. So the

relation will be of higher order. For instance, consider an event such as a disease outbreak; it is needed to define the relation between the disease name, the location of the outbreak, the number of affected or killed people and the date of the outbreak. Another example of multi-way relations is defined as Semantic Role Labeling [Gildea and Jurafsky, 2001] where the goal is to find arguments of a predicate in a given sentence. For example, for the predicate *"accept"* in the sentence *"He accepted the gift from his father"*, the acceptor is *"he"* and the *"the accepted"* is *"the gift"*.

- *Adjectives describing entities* are also important in some applications of IE as explained in Section 7.1.1. For example, given an entity type such as restaurant name, we need to extract part of a blog post or a web page that is about the given entity. Then, we might need to infer if there is any critic (positive or negative) about the given name in the extracted data. This kind of processing is mostly considered in *Opinion Mining* [Pang and Lee, 2008] and is not covered in this thesis.

- *Lists and tables* also contain information that could be considered in an IE task [Embley et al., 2006]. However, they are out of the scope of this document.

Among all of these four types of data, we focus on entities and relations between them in this research; although some of the proposed modules may apply for other type of information.

7.2 Machine Learning

Machine Learning (ML) is a group of models and algorithms that are designed to learn from sampled data or to mine it to discover hidden information. There are various learning algorithms which we can review in two main groups. On one side, we have supervised models that require labeled training data as an input. They need to learn from the training data in order to model it; the learned model is then used to predict new unknown data. On the other side, there are algorithms that are designed to model (raw) data without any supervision, which are called unsupervised models. These models can be used to discover common patterns or structures in the data. In this case, the goal is to mine the data with minimum assumption about it. Another group of techniques models the problem by using a combination of labeled data (common with supervised ML) and raw data (similar to unsupervised ML). These techniques are called semi-supervised ML. We will not cover them in this chapter since they are not used by the models that are proposed in this thesis.

The two following subsections briefly describe some supervised and unsupervised techniques useful to understand our work. The last section is dedicated to the evaluation approaches of these techniques.

7.2.1 Supervised Techniques

All Machine Learning algorithms that need training data to model a problem are considered in this category. Since the training data are mostly prepared by humans, they are referred either as labeled or annotated data. In this thesis, whenever we refer to training data, it means also annotated data and vice versa. Many successful systems have used supervised ML; yet it has a major drawback. Indeed, preparing annotated data is a bottleneck of these algorithms because it is hard and expensive to build accurate training data. Moreover, moving from one domain to another with a trained supervised algorithm tends to decrease the system performance. Here, it often requires a new training data of the new domain.

In the following, we review two main algorithms which are either used or important in the model proposed in Chapter 8: k-Nearest Neighbors and Support Vector Machine.

7.2.1.1 K-Nearest Neighbors

K-Nearest Neighbors (KNNs) [Cover and Hart, 1967] is a supervised Machine Learning algorithm that can classify objects based on their k nearest neighbors in the training data. Three main components have to be defined: the number of neighbors k, the distance function and the voting process. KNN is based on a voting among the k-nearest neighbors, defined according to the distance chosen, of the new object . It can bias the voting toward more frequent objects, specially if all nearest neighbors have the same impact on the final result. A weighted voting model, for example based on the distance to other objects, can help to avoid biased voting.

KNN is an instance-based learning and the complexity can grow with the data size. For example, if there are n objects in the training data, in the worse case, the complexity of classifying a new instance is $O(n)$ which can be very costly depending of the distance used. However, a clever implementation (e.g. tree-based) can reduce the computation time [Daelemans et al., 1997]. Storing objects in an ordered tree data structure reduces the time to find the nearest neighbor objects.

The proper value for k in KNN depends on the data. A good value can be estimated with some heuristic algorithms such as cross-validation, assuming that the best value for k yield the best result in this evaluation technique. An example

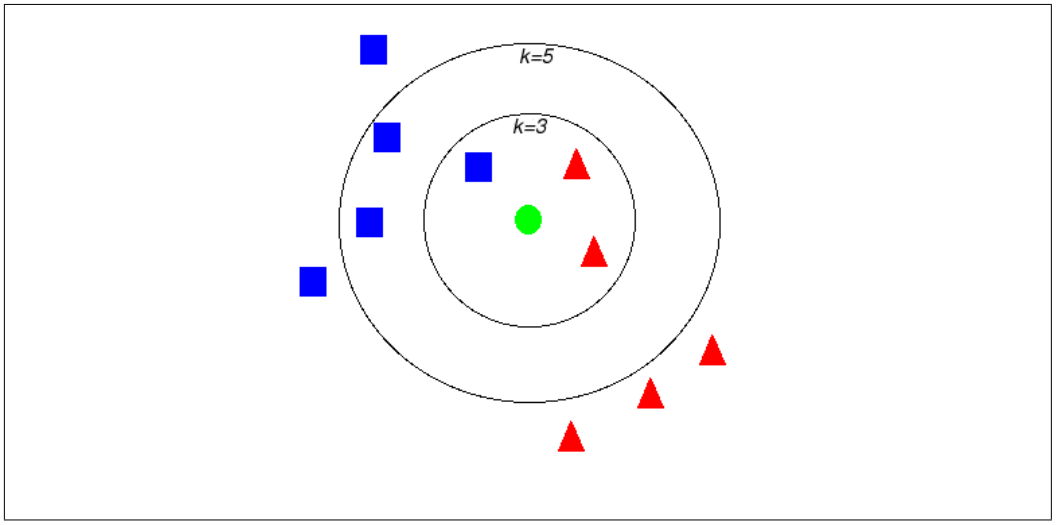

Figure 7.1: KNN example: With $k = 3$, the green circle is in the red triangle class but with larger $k = 5$, the final class is blue square.

of KNN is depicted in Figure 7.1 so that increasing k from 3 to 5 changes the result. For $k = 3$, the final class for *green circle* is *red triangle*, but with $k = 5$, *blue square* is the result.

Usually, the Euclidean distance is used as the distance metrics in KNN (if features are defined in a vectorial space). However other metrics such as the overlap metric (or Hamming distance), where the distance between two strings is the minimum number of substitutions required to change one string into another, are also used with this algorithm [Kolbe et al., 2010]. We introduce a new distance (similarity) metric in Chapter 8 based on the Language Modeling (explained in section 7.3.2.1) to classify relations based on a KNN model.

7.2.1.2 Support Vector Machine

Support Vector Machine (SVM) [Cortes and Vapnik, 1995] is a supervised Machine Learning algorithm which has shown promising empirical results in many practical applications such as handwriting recognition [Bahlmann et al., 2002] and text classification [Joachims, 1998]. Generally, a Support Vector Machine constructs one or more hyperplanes in a high dimensional space in order to classify objects (represented as points in this space). Intuitively, a good hyperplane has the largest distance from the nearest training data point of any class.

In the literature, for a two-class problem, data are defined linearly separable (see Figure 7.2) if there exists a function based on the linear combination of data features to separate the data into two classes. In other words, two sets of points

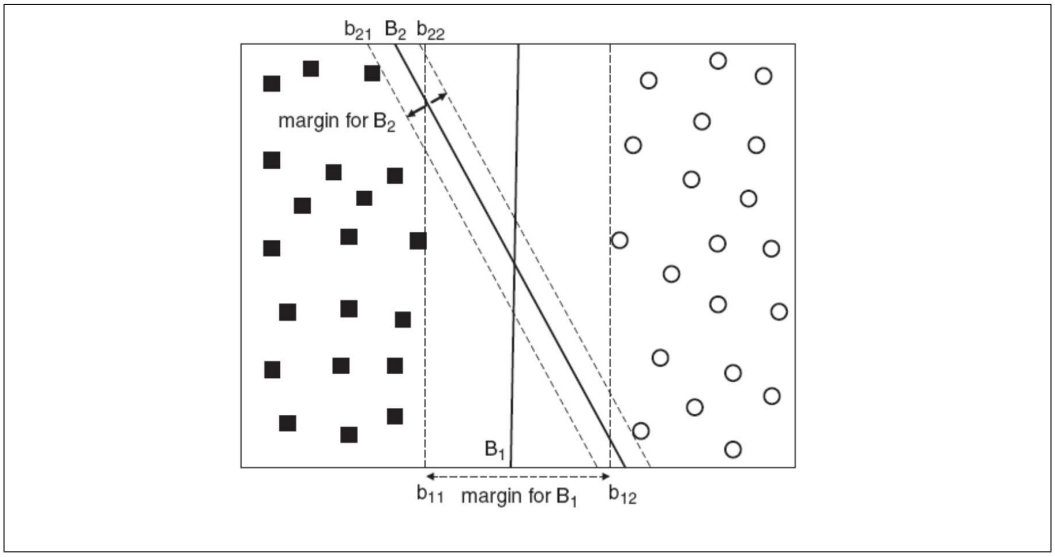

Figure 7.2: Possible hyperplanes for classifying a linearly separable data set [Tan et al., 2006]. B_1 is a better hyperplane than B_2 because it maximizes the separation of the classes.

are linearly separable, if a single line can separate them; otherwise the data are not linearly separable. Assuming a dataset D with n points (in Equation 7.1), let x_i be a p-dimensional real vector and $y_i \in \{+1, -1\}$ as the "class" of x_i. We are looking for a hyperplane to divide the data into two parts with $y = +1$ and $y = -1$.

$$D = \{(x_i, y_i)|x_i \in \mathbb{R}, y_i \in \{-1, +1\}\}_{i=1}^{n} \tag{7.1}$$

Many lines can separate linear separable data, but we are looking for the line with the maximum distance from both parts. For example, in Figure 7.2, B_1 is a better hyperplane than B_2 because it maximizes the margin or the separation of classes. This is done by complex computation of scalar products on more complex functions called "kernels".

SVMs often show better results in practice when other classification algorithms perform poorly. Furthermore, this algorithm works for both linear and non-linear separable data. When the data are not linearly separable, we can define kernels to map the data to higher dimensions where the data are linearly separable. Since the SVM training is relatively fast (depending on the kernel used), it is easy to use it for problems with high dimensional feature space. These properties make the SVM popular in many applications such as text classification. However, it still needs parameter tuning which is not a trivial task [Chapelle et al., 2002] and requiring to choose the kernel suited to the data.

7.2.2 Unsupervised Techniques

Unsupervised Machine Learning algorithms are used in Chapter 9 and 10. Unsupervised learning means that there is no human experts who label the training data. However, we might define some unsupervised algorithms based on our knowledge about the data, but we do not need to have labeled data for this kind of algorithm. Among different models, we explain Clustering as the most common form of unsupervised learning in the next section. Then, in Section 7.2.2.2, we review k-means as a popular clustering technique. Markov Clustering algorithm is explained in Section 7.2.2.3 since we use it in this thesis. Finally, some similarity functions are briefly explained in the last section.

7.2.2.1 Clustering: general considerations

A clustering algorithm puts a set of objects into subsets or *clusters* based on the similarity (or distance) between them. Intuitively, the goal is to have more similarities between the objects within a cluster and less similarities between objects in different clusters. However, more constraints are defined for good clustering algorithms; they are explained in Section 7.2.3.2.

Different clustering algorithms are proposed in the literature. They can be considered in two main groups [Manning et al., 2008]. *Flat clustering*, as the first group, creates a flat set of clusters without explicit relations between clusters. Instead, *hierarchical clustering* builds a set of clusters in a hierarchical structure in which the relations between the clusters are defined as a tree-based structure.

Assuming singleton clusters (each cluster has only one member), different strategies are defined to merge clusters in a hierarchical model including single-link, complete-link, average and centroid clustering [Manning et al., 2008]. In the single-link (single-linkage) clustering, the similarity between two clusters is computed based on the similarity between most similar members (Figure 7.3(a)). The merging decision is local since it depends on the areas where the clusters are close. The overall structure of the cluster will not affect the final decision. On the contrary, in the complete-link (complete-linkage) clustering, the similarity between two clusters is the similarity between most dissimilar members (Figure 7.3(b)). This method does not have any local problem, but still has a problem of sensitivity to outliers. In the average clustering, the similarity is defined as the average similarity between all members of two clusters. Similarity between two clusters in centroid clustering is based on the similarity of their centroids.

Additionally, another important difference between clustering algorithms is about Soft versus Hard clustering [Manning et al., 2008]. In *Soft clustering*, each cluster can have all objects as its members but with a degree of membership. On the contrary, in *Hard clustering*, each object belongs to only one cluster. Tan et al. [2006] proposed to distinguish three groups of clustering algorithms,

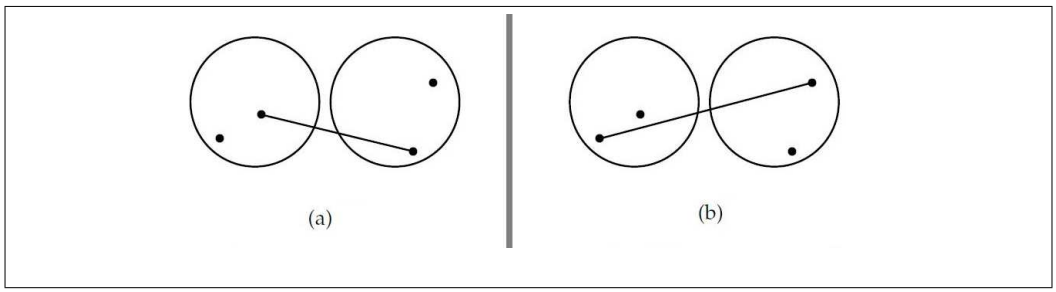

Figure 7.3: Two different notions of cluster similarity [Manning et al., 2008]. (**a**) *Single link*: similarity of the most similar members (**b**) *Complete link*: similarity of the most dissimilar members

Exclusive versus *Overlapping* versus *Fuzzy*. In the case of having only one cluster for each object, the algorithm is *Exclusive clustering*. If all objects belong to all clusters but with degree of membership, it is *Fuzzy clustering*. In some cases, we only have some common objects between clusters, which is the definition of *Overlapping clustering*. All of the clustering algorithms that we review in this chapter or use in this thesis are Flat, Hard and Exclusive clustering.

In the following, two popular clustering algorithms, k-means and Markov Clustering are explained. These clustering models are based on the similarity among objects, calculated by a similarity function. Then, some similarity functions are reviewed after that. We propose different similarity functions in Chapter 8 and 9 as an important contribution of this thesis.

7.2.2.2 K-means

K-means clustering is the most used flat and hard clustering algorithm. The goal is to minimize the distance of objects in a cluster from the centroid.

For a given set of n objects with $X = \{x_1, ..., x_l\}$ where x_i is the feature vector for the i^{th} object, the goal is to split the objects into k subsets $S = \{s_1, ..., s_k\}$ in a way that the Sum of Squared Distances (SSD) of objects from the centroid in each cluster is minimized (see Equation 7.2, μ_i is the centroid of s_i and δ is the distance chosen).

$$\underset{s}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x_j \in S_i} \delta(x_j - \mu_i)^2 \tag{7.2}$$

Different implementations of k-means algorithm has been proposed in the literature. All of them have two main steps in common: the *assignment* and *update* steps. Given an initial set of k centroids $\{\mu_1, ..., \mu_k\}$, the algorithm iterates over the following steps:

- *Assignment step*: assign each object to the cluster with the closest μ. In this step, each object has to go to only one cluster.

- *Update step*: calculate the new centroid for each cluster.

One important property of any algorithm is its time complexity. The assignment step computes $K \times N$ (K is the number of clusters and N is the size of feature vector) distances and the complexity of each distance computation is $O(M)$ (M is the number of features). So the complexity is $O(KNM)$. For the update step, the complexity is $O(NM)$. Considering the number of iterations as I, the overall k-means complexity is $O(IKNM)$ which makes the complexity of k-means linear to all of its parameters.

One way to make the algorithm faster is to consider centroid-object similarity instead of object-object similarity. In this way, in each assignment step, we only need to compute the distance of the new object from the centroid of each cluster. The complexity of the assignment step is $O(IK(N-K)M)$ which is better than the naive k-means. The algorithm is even faster if we use *medoids* instead of *centroids*. The medoid of a cluster is defined as the object that is closest to the centroid. Since, object vectors are sparse, distance computation can be faster. Moreover, all the needed computations can be done for all at the beginning with this approach

Although k-means is simple to implement, it needs k to be determined in advance which is not a trivial task. A bad estimation of k leads the algorithm to output meaningless results.

7.2.2.3 Markov Clustering

If we model the data as a graph, considering objects as nodes and the similarity (or distance) between them as edges between the nodes, we can use graph clustering algorithms to split the objects into subsets based on their similarities (or distances) to each other. Among all different graph clustering algorithms, we explain Markov Clustering algorithm as an example which is used in two chapters of this thesis (chapters 9 and 10).

Markov Clustering (MCL) [van Dongen, 2000a] is a graph clustering algorithm in which the goal is to have many vertices within the clusters and few between the clusters. It means that if a walk start from a node and goes randomly to connected nodes (Random Walk [Harel and Koren, 2001]), it is more likely to stay within the same cluster than to go to other clusters. The Random Walk on a graph is defined based on *Markov Chains* which is a mathematical model of the transitions from one state to another, among a finite number of possible states (e.g. in a graph). The probability of moving from one state to another

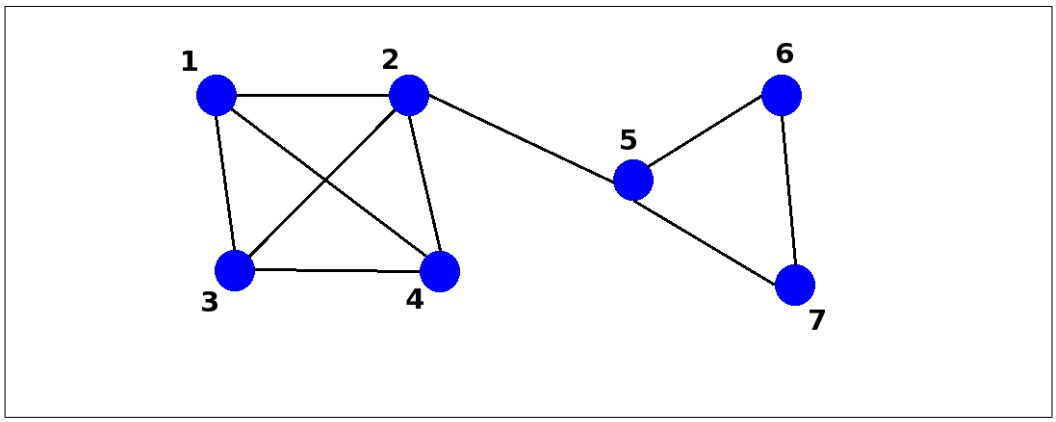

Figure 7.4: A simple graph with unweighted edges

state depends only on the current state, but not on the past and the future states (Markov property).

Considering the simple graph in Figure 7.4, we assume all vertices are equal (i.e. unweighted) for simplicity. A random walk from node 1 has 33% chance to go to each of the nodes 2, 3 and 4 and no chance to go to nodes 5, 6 and 7. A random walk from node 2 has 25% chance to go to 1, 3, 4 and 5 and no chance for other nodes in the graph.

The transition matrix of the graph in Figure 7.4 is defined in Equation 7.3 where, for each edge between two nodes, there is a non-zero value (one) in the matrix.

$$TransitionMatrix = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \tag{7.3}$$

In order to build the probability matrix, which is needed by the Random Walk, we need to normalize the matrix in columns; this is performed in Equation 7.4, by dividing each cell value by the sum of the cell values in the column.

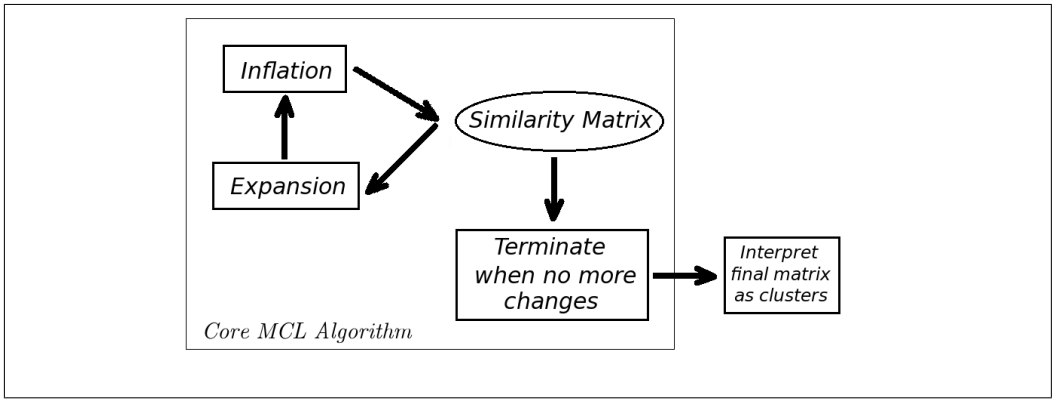

Figure 7.5: Markov Clustering Algorithm [van Dongen, 2000a] is based on the iteration between *inflation* and *expansion*. It terminates when nothing changed in the similarity matrix.

$$ProbabilityMatrix = \begin{bmatrix} 0 & 1/4 & 1/3 & 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/3 & 1/4 & 0 & 1/3 & 0 & 0 & 0 \\ 1/3 & 1/4 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 1/2 & 0 \end{bmatrix} \tag{7.4}$$

Generally, for a graph clustering algorithm, the similarity between nodes (objects) is defined as the weight of the corresponding edge between them. In the case of no similarity between two objects, there is no edge between them. MCL uses a similarity matrix which is similar to the transition matrix but applies a weighted similarity.

$$Sim_{X_n} = \begin{bmatrix} s_{11} & s_{12} & \dots & s_{1n} \\ s_{21} & s_{22} & \dots & s_{2n} \\ \dots & \dots & \dots & s_{ij} \\ s_{n1} & s_{n2} & \dots & s_{nn} \end{bmatrix} \tag{7.5}$$

In the similarity matrix (Equation 7.5), each entry cell s_{ij} is the similarity between node i and node j in the graph. Markov Clustering algorithm is defined as an alternation between two main operations, *inflation* and *expansion* (Figure 7.5).

The **expansion operation** is a simple matrix multiplication operation which is shown in Equation 7.6 for the matrix in Equation 7.5. This operation makes

new paths between nodes.

$$Sim^*_{X_n} = \begin{bmatrix} \sum_{i=1}^{n} s_{1i}.s_{i1} & \sum_{i=1}^{n} s_{1i}.s_{i2} & \dots & \sum_{i=1}^{n} s_{1i}.s_{in} \\ \sum_{i=1}^{n} s_{2i}.s_{i1} & \sum_{i=1}^{n} s_{2i}.s_{i2} & \dots & \sum_{i=1}^{n} s_{2i}.s_{in} \\ \dots & \dots & \dots & \dots \\ \sum_{i=1}^{n} s_{ni}.s_{i1} & \sum_{i=1}^{n} s_{ni}.s_{i2} & \dots & \sum_{i=1}^{n} s_{ni}.s_{in} \end{bmatrix} \tag{7.6}$$

The **inflation operation** models the contraction of a flow between nodes. It becomes thicker for edges with multiple paths and thinner for other edges. It is defined as the similarity matrix power to I (inflation number). The Inflation number is the most important parameter that can change the MCL results. For larger I, the algorithm can converge faster but may cause staying in a local minimum. Finding a good Inflation rate is essential to have reasonable results. Medvés et al. [2008] proposed a dynamic inflation rate. The authors show that a time varying inflation rate can help to have better results. The inflation rate begins with a large inflation rate in the first iteration to find the most distinguished clusters and then reduce the rate for a finer tuning of the clusters in later iterations.

$$(Sim_{X_n})^I = \left(\begin{bmatrix} s_{11} & s_{12} & \dots & s_{1n} \\ s_{21} & s_{22} & \dots & s_{2n} \\ \dots & \dots & \dots & s_{ij} \\ s_{n1} & s_{n2} & \dots & s_{nn} \end{bmatrix} \right)^I \tag{7.7}$$

MCL has some advantages compared to k-means. First of all, there is no need to know the number of clusters at the beginning which is needed by k-means. Second, it is a simple algorithm that shows good results for different graph clustering problems [van Dongen, 2000a; Wang et al., 2011]. The complexity of MCL depends on its two main operations: the expansion and the inflation. The expansion operation is defined as multiplication of two $n \times n$ matrices so the complexity is $O(n^3)$. Since the complexity of matrix normalization is $O(n^2)$, we can consider the bigger complexity for this operation. The complexity of inflation operation is $O(n^2)$. So, the overall complexity of MCL is the higher one, $O(n^3)$. However, Stother [2010] has shown that the possibility of multiplying two $n \times n$ matrices with the complexity of $O(n^{2.8074})$ which is slightly better. Moreover, because of the matrix sparsity, the complexity can be reduced to $O(\mathrm{n}\ k^2)$ by using a sparse matrix implementation [van Dongen, 2000b].

7.2.2.4 Similarity Functions for Vectors

Similarity function is one of the essential parameters of any clustering algorithms [Manning et al., 2008]. In this thesis, different similarity functions are proposed

Similarity function		Definition
$Cosine(A,B)$	$=$	$\frac{\sum_{i=1}^{n} a_i * b_i}{\sqrt{\sum_{i=1}^{n} (a_i)^2} * \sqrt{\sum_{i=1}^{n} (b_i)^2}}$
$Jaccard(A,B)$	$=$	$\frac{\lvert A \cap B \rvert}{\lvert A \cup B \rvert}$

Table 7.1: Two classical similarity functions, *Cosine* for weighted vectors and *Jaccard* for unweighted vectors

as main contributions (see chapters 9 and 10). In order to show the importance of our proposed similarity functions, we compare them experimentally, in chapters 9 and 10, to classical similarity functions.

Considering two objects defined with two feature vectors $A = \{a_1, ..., a_n\}$ and $B = \{b_1, ..., b_n\}$, different classical similarity functions are defined in Table 7.1. All similarity functions have two main properties, symmetry and positivity. To be more explicit, if $sim(A, B)$ is the similarity between two objects A and B, then the following properties are always true [Tan et al., 2006]:

- $sim(A, B) = 1$ only if A=B. (Positivity)
- $sim(A, B) = sim(B, A)$ for all A and B. (Symmetry)

Clearly, all features in the feature vector do not have the same contribution to the similarity. Among the two similarity functions in Table 7.1, *Cosine* can use weighted feature vectors. Since a simple feature frequency does not have enough information about the feature importance, different weighting schemes are used to distinguish important features and make them more effective for objects similarity calculation. Some of these weighting schemes are defined in Section 7.3.3, in the context of text representation for NLP.

7.2.3 Evaluation

Different Machine Learning algorithms are proposed in the literature and the results need to be evaluated in order to have better comparisons between them. In the following, we review some evaluation techniques for supervised and unsupervised Machine Learning algorithms that are used in our works.

7.2.3.1 Classification Evaluation

In supervised Machine Learning, once the model has been constructed, it can be applied on an unseen test data to predict the class labels. The accuracy or error rate serves as the evaluation measure to estimate the performance of a model. In this section, among all different evaluation techniques [Tan et al., 2006], we review some of them that are used in this thesis, including cross-validation and leave-one-out validation.

K-fold cross-validation is an evaluation technique in which the original data is randomly partitioned into k folds. A single fold is considered as the validation data and the remaining $k-1$ folds are used for training. The validation is then repeated for all k folds in which each fold is used only once for the validation. Then, the k results can be averaged to estimate the final evaluation. It has been shown that ten-fold cross-validation is enough in most of the cases [Kohavi, 1995].

Leave-One-Out validation (LOO validation) is a special k-fold cross-validation where the k is equal to the number of objects in the training data. Clearly, this method is very expensive, because it needs many repetitions of training which makes this method impossible to be used with large scale data sets. Kohavi [1995] compared ten-fold cross-validation and LOO validation on a large scale data classification problem with different algorithms. Compared to the LOO validation which is expensive computationally, ten-fold cross-validation always assess results as good as LOO techniques.

K-fold cross-validation is useful to compare the performance of two different approaches. However, sometimes a model shows good results compared to the others by this evaluation but it can not show similar performance for unseen data. In some applications (e.g. Protein-Protein Interaction extraction), using an external data for evaluation that did not exist in the training data can help to have better evaluations.

7.2.3.2 Clustering evaluation

Typically, the goal in any clustering algorithm is to have more similarities between objects within the same cluster and less similarities between objects in different clusters. This is an *internal criterion* for the evaluation of a clustering. But the high score for internal evaluation does not mean better results in an application. An alternative to the internal criterion is a direct evaluation with a ground-truth based on *external metrics*. Ground-truth or gold standard, which is produced by domain experts, can show how effective the clustering is to a specific application but is of course expensive to build.

In the following, we explain five external evaluation metrics including Purity, Inverse Purity, F-measure, Rand Index and Adjusted Rand Index which are used in this thesis. All of these metrics are defined by considering $C = \{c_1, c_2, ..., c_N\}$

as a set of clusters (thus, c_i is a set of objects) and $L = \{l_1, l_2, ..., l_M\}$ as a set of classes in the ground-truth, where M is the number of classes in the ground-truth and N is the number of clusters in the final clustering result.

Purity [Zhao and Karypis, 2001] is a simple and transparent evaluation metric which is defined as the average of maximal precision of each cluster which is defined in Equation 7.8.

$$\textbf{Purity}(C, L) = \frac{1}{N} \sum_{i=1}^{N} max_j \text{ Precision}(C_i \cap L_j), 1 \leq j \leq M \tag{7.8}$$

$$\text{Precision}(C_i \cap L_j) = \frac{|C_i \cap L_j|}{|C_i|} \tag{7.9}$$

where $|C_i \cap L_j|$ is the number of common objects between C_i and L_j, $|C_i|$ is the total number of objects in C_i.

To calculate the Purity, each cluster is assigned to its the most frequent class in the ground-truth and then the precision of this assignment is the precision for that cluster (see Equation 7.9). Finally, the average of all computed precisions is the Purity.

Inverse Purity [Amigo et al., 2009] is another metric that can reward grouping similar items together (Equation 7.10).

$$\textbf{Inverse Purity}(C, L) = \frac{1}{M} \sum_{j=1}^{M} max_i \text{ Precision}(C_i \cap L_j) \tag{7.10}$$

The highest value for Purity occurs when each cluster has only one member. Purity cannot reward grouping similar members together. Similarly, Inverse Purity value is equal to one in the case of having all members in one cluster. Yet, a more robust metric is needed to combine the advantages of both Purity and Inverse Purity.

F-measure [van Rijsbergen, 1974] is a weighted average of Purity and Inverse Purity benefiting from the advantages of both metrics. It is defined in Equation 7.11.

$$F_\beta - measure(C, L) = \frac{(1 + \beta^2) \times Purity(C, L) \times \text{Inverse Purity}(C, L)}{\beta^2 \times Purity(C, L) + \text{Inverse Purity}(C, L)} \tag{7.11}$$

Equation 7.11 with $\beta = 1$ is the harmonic mean of Purity and Inverse Purity.

Rand Index [Rand, 1971] is a measure of similarity (agreement) between two data objects (e.g. a ground-truth and a clustering). Assume that True Positive (TP) decision assigns two similar objects to the same cluster and True Negative (TN) allocates two different objects to different clusters. Similarly, a

Class	v_1	v_2	...	v_c	**Sums**
u_1	n_{11}	n_{12}	...	n_{1c}	a_1
u_2	n_{21}	n_{22}	...	n_{2c}	a_2
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
u_r	n_{r1}	n_{r2}	...	n_{rc}	a_r
Sums	b_1	b_2	...	b_c	

Table 7.2: Overlap between U and V

False Positive (FP) decision assigns two dissimilar objects to the same cluster and a False Negative (FN) decision assigns two similar objects to different clusters. Then, *Rand Index*, which measures the percentage of decisions that are common, can be defined by Equation 7.12.

$$RI = \frac{TP + TN}{TP + FP + FN + TN} \tag{7.12}$$

Rand Index has a value between 0 and 1, where 0 indicates that two clusterings do not agree on any pair of objects and 1 shows that both clusterings are exactly the same. The random agreement between two clusterings is not a constant (zero) in Rand Index. So, it has to be corrected in order to have better evaluation measure.

Adjusted Rand Index [Hubert and Arabie, 1985] (ARI) is defined as the correction by chance version of Rand Index. Given a set of n elements $S = \{O_1, ..., O_n\}$ and two partitions of S to be compared, say $U = \{u_1, ..., u_r\}$ and $V = \{v_1, ..., v_c\}$, the overlap between U and V is summarized in Table 7.2, where n_{ij} is the number of common objects between two partitions u_i and v_j.

The Adjusted Rand Index is defined in Equation 7.13.

$$ARI = \frac{Index - ExpectedIndex}{MaxIndex - ExpectedIndex} \tag{7.13}$$

where

$$Index = \sum_{ij} \binom{n_{ij}}{2}$$

and

$$ExpectedIndex = \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}$$

and

$$MaxIndex = \frac{1}{2}(\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2})$$

ARI value is between 0 and 1. The ARI score for random clustering is near to zero which makes this metric more suitable to compare two clustering algorithms. Santos and Embrechts [2009] have shown that ARI can help to improve the feature selection for data with a high number of features.

We review some evaluation metrics that are popular to evaluate clustering algorithm in this sub-section. Among all of them, ARI is the only metric that can take into the account the random agreement between two clusterings. This the main reason that we reported our results only based on ARI in this thesis.

7.3 Natural Language Processing Techniques

Natural Language Processing (NLP) was introduced by Turing [1950] when he designed a test (Turing test) to investigate the machine intelligent behavior. The goal of the test was about the answer to this question: are we able to design a computer program to have a real-time written conversation with a human so that (s)he is unable to find the truth (if the talk is with machine or human) based on the exchanged questions and answers. Currently, NLP is about processing of natural language contents such as text, speech. For each of NLP applications (e.g. Machine Translation, Text Classification, Text retrieval), different models of processing are proposed including syntactic and statistical analysis. In this section, we first review some models on syntactic analysis and follow the review with some statistical analysis. In these reviews, we only cover the concepts and techniques that are either used in our proposed models or used for comparison.

7.3.1 Deep and Shallow Linguistic Analysis

Different levels of information are used for analyzing Natural Language, they can be considered in two groups based on the amount of analysis that is needed. The analysis can begin from the word forms (separated by a space, a comma or other separators) to the syntactic and semantic analysis of a sentence. Of course, setting such a formation among the different processings is artificial and subjective but taking into account the complexity and a priori knowledge of NLP techniques is of outmost importance when dealing with robustness and potability. Thus, in the next two sections, some techniques are presented, according to the level of the analysis, first shallow and then deep linguistic analysis.

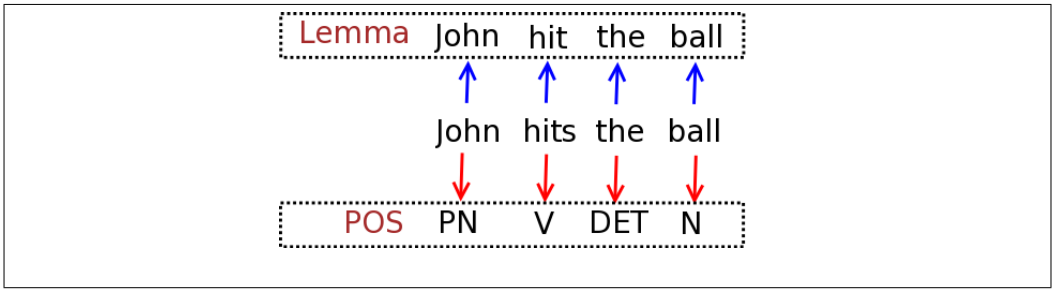

Figure 7.6: POS tags for "John hits the ball"

7.3.1.1 Shallow analysis

The simplest way of analyzing a sentence is to consider the word forms (word occurrences separated by a space, a comma or other separators). Even if they are considered as a set of words without any order, they still can be useful in some applications. But most of the time, more linguistic information is used to solve a problem like text classification or information extraction from text. In these cases, Part-of-Speech tags as a shallow linguistic analysis of the text can be an option.

Part-of-Speech (POS)
POS tagging is a task of assigning a grammatical tag (e.g. noun, verb) to a word in a sentence based on the word and its surrounding. Part-of-speech tagging is more difficult than extracting a word and its POS from a dictionary, because some words might have more than one tag. For example, a simple word "dogs", which is usually the name of an animal, can be also a verb in "*The sailor dogs the barmaid.*", where "dogs" means fasten securely. As an example, POS tags for a sample sentence are given in Figure 7.6. In general, a POS tagger can distinguish 150 POS tags for English but in practice the number of POS tags are less. For example, TreeTagger [Schmid, 1994] can recognize 33 POS tags for French text and 36 POS tags for English. The size of the tag sets depends on the corpus that is used for training and the application. For example, the Brown Corpus [Kučera and Francis, 1967] uses 82 POS tags and the Penn Treebank [Marcus et al., 1993] has 48 POS tags. Another famous textual corpus, the British National Corpus [Clear, 1993], defined 57 tags as its basic tag set. Table 7.3 shows the tag set of TreeTagger [Schmid, 1994] for French that is used in our experiments.

Currently, POS taggers can predict the grammatical tags for words almost as well as humans and are also available for different languages. Even if a POS tagger is not available for a language, it can, most of the time, be easily built by training one of the freely available taggers such as Brill tagger [Brill, 1992] or more modern approaches (MaxEnt [Ratnaparkhi, 1996], HMM [Kupiec, 1992],

POS	Explanation
ABR	abreviation
ADJ	adjective
ADV	adverb
DET:ART	article
DET:POS	possessive pronoun (*ma, ta, ...*)
INT	interjection
KON	conjunction
NAM	proper name
NOM	noun
NUM	numeral
PRO	pronoun
PRO:DEM	demonstrative pronoun
PRO:IND	indefinite pronoun
PRO:PER	personal pronoun
PRO:POS	possessive pronoun (*mien, tien, ...*)
PRO:REL	relative pronoun
PRP	preposition
PRP:det	preposition plus article (*au, du, aux, des*)
PUN	punctuation
PUN:cit	punctuation citation
SENT	sentence tag
SYM	symbol
VER:cond	verb conditional
VER:futu	verb futur
VER:impe	verb imperative
VER:impf	verb imperfect
VER:infi	verb infinitive
VER:pper	verb past participle
VER:ppre	verb present participle
VER:pres	verb present
VER:simp	verb simple past
VER:subi	verb subjunctive imperfect
VER:subp	verb subjunctive present

Table 7.3: POS tag list in TreeTagger trained for French language

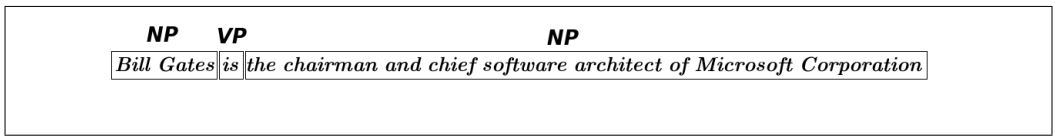

Figure 7.7: Chunks in a sample sentence

etc.).

Chunk [Abney, 1991]
POS tags can give linguistic information at the word level but cannot specify the constituents and phrases in a sentence. Chunking or shallow parsing is a solution for those problems that require phrases but do not need to have a full syntactic analysis as explained in Section 7.3.1.2. Chunking is an analysis of a sentence which identifies constituents such as noun or verb groups, but does not specify their internal structure, nor their role in the main sentence. For example, in Figure 7.7, we have three chunks including two Noun Phrases and one Verb Phrase. In this example, there is no detail about the second Noun Phrase which is a combination of nine words. This analysis can help us to identify the most important syntactic components of a sentence without going into a detailed analysis.

7.3.1.2 Deep analysis

Analyzing a sentence as a sequence of words or chunks is not enough for some applications. We might need the relations between different parts of a sentence to know, for example, subject-object or subject-predicate relations in the sentence. Every syntactic theory contains either Phrase Structure analysis or Dependency analysis or possibly both.

Phrase Structure Grammar
Chomsky [2002] originally introduced the term Phrase Structure Grammar (PSG) where the grammar is defined based on some phrase structure rules. PSGs are also called Context-Free Grammars [Chomsky, 1956], but changed later, so that there are some Context Sensitive Grammars which are defined under the PSG category.

For example, Figure 7.8 is a phrase structure parse tree for the sentence "*This is an example of constituency grammar*". This tree is generated by using some phrase structure rules listed in Table 7.4. Phrase structure rules are generally in the form of $A \rightarrow$ B C which means that the constituent A can be separated into two sub-constituents B and C. In phrase structure rules, we define NP, VP, ... as non-terminal and all words as terminal symbols.

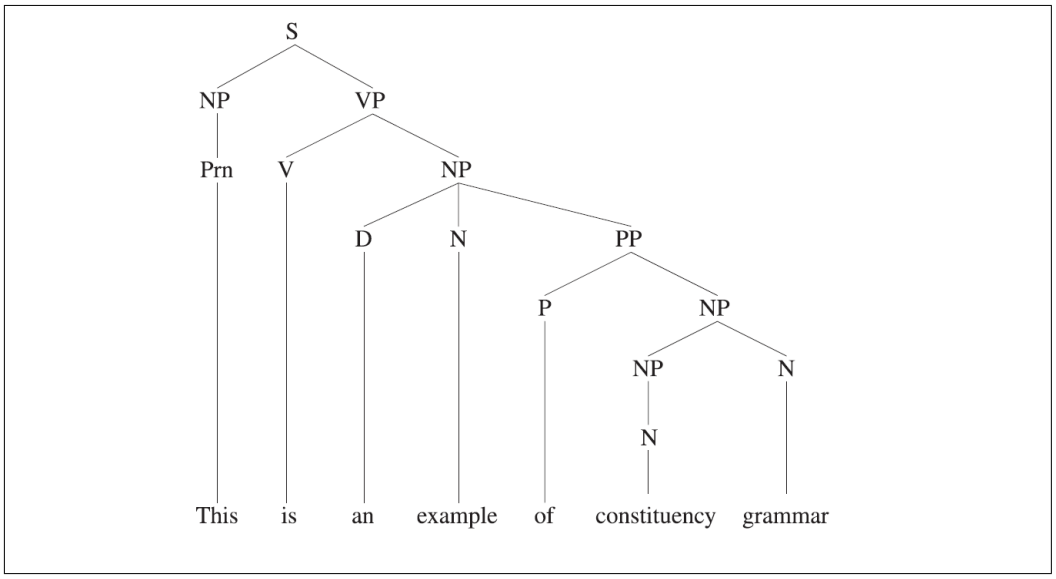

Figure 7.8: A constituency tree [Covington, 2001] for "This is an example of constituency grammar", where non-terminal symbols build intermediate nodes and terminal symbols are leaf nodes.

No	Rule
1	S → NP VP
2	NP → Prn
3	NP → D N PP
4	NP → NP N
5	NP → N
6	VP → V NP
7	PP → P NP

Table 7.4: Phrase Structure Grammar to parse "*This is an example of constituency grammar*"

There are some ambiguities that make PSG difficult to be used even for formal sentences. For example, Propositional Phrase (PP) attachment is the most common source of ambiguity in natural language [Collins and Brooks, 1995]. In the sentence "*Pierre Vinken, 61 years old, joined the board as a nonexecutive director*", the PP "*as a non-executive director*" can either attach to NP "*the board*" or to the VP "*joined*" which can give us two different parse trees [Collins and Brooks, 1995]. In addition to the mentioned ambiguity problem with PSG, it cannot work properly with informal text, such as transcribed text (informal sentences with lots of noisy tokens and oral disfluencies). It also has problem with recently most used informal text in Internet chat or micro blogging systems such as Tweeter. In these cases, shallow linguistic information which is more robust to noisy data and informal text can help, for example, in Information Extraction applications.

Dependency Grammar

The concept of word-to-word dependency is found in traditional Latin, Arabic and Sanskrit grammars [Covington, 2001]. Dependency Grammar (DG) is a class of syntactic theories that are based on such dependency relations between words. Whenever two words are connected by a dependency relation, one of them is the **head** and the other is the **dependent**. Essentially, a dependency relation can be represented as an arrow from the head to the dependent. The arrows can be labeled with the actual dependency nature (e.g. subject, object) or not, depending on the linguistic framework. Dependency Grammars build Dependency Trees (DT). An example of Dependency Tree (DT) is shown in Figure 7.9 where the main verb is the root of the tree and the other words are the dependents of the verb. Dependency trees have also constituents (phrases) which are words with their dependents (and dependents of dependents). Gaifman [1965] proved that "Every dependency grammar has a *naturally corresponding* Phrase Structure Grammar but not vice versa."

A Dependency Tree is a directed acyclic graph and thus has some properties which are listed below:

- **Antitransitive:** if $A \rightarrow B$ and $B \rightarrow C$, then $A \not\rightarrow C$
  (e.g. *a very smart student*: *student* $\rightarrow$ *smart* $\rightarrow$ *very* but *student* $\not\rightarrow$ *very*)
- **Antisymetrics:** if $A \rightarrow B$, then $B \not\rightarrow A$
  (e.g. *red* $\rightarrow$ *book* $\neq$ *book* $\rightarrow$ *red*)
- **Antireflexive:** if $A \rightarrow B$, then $A \neq B$
  (e.g. *red* $\rightarrow$ *book* then *book* $\neq$ *red*)

For text processing applications, we might need either deep or shallow linguistic analysis or in some cases both. But there are some limitations for utilizing

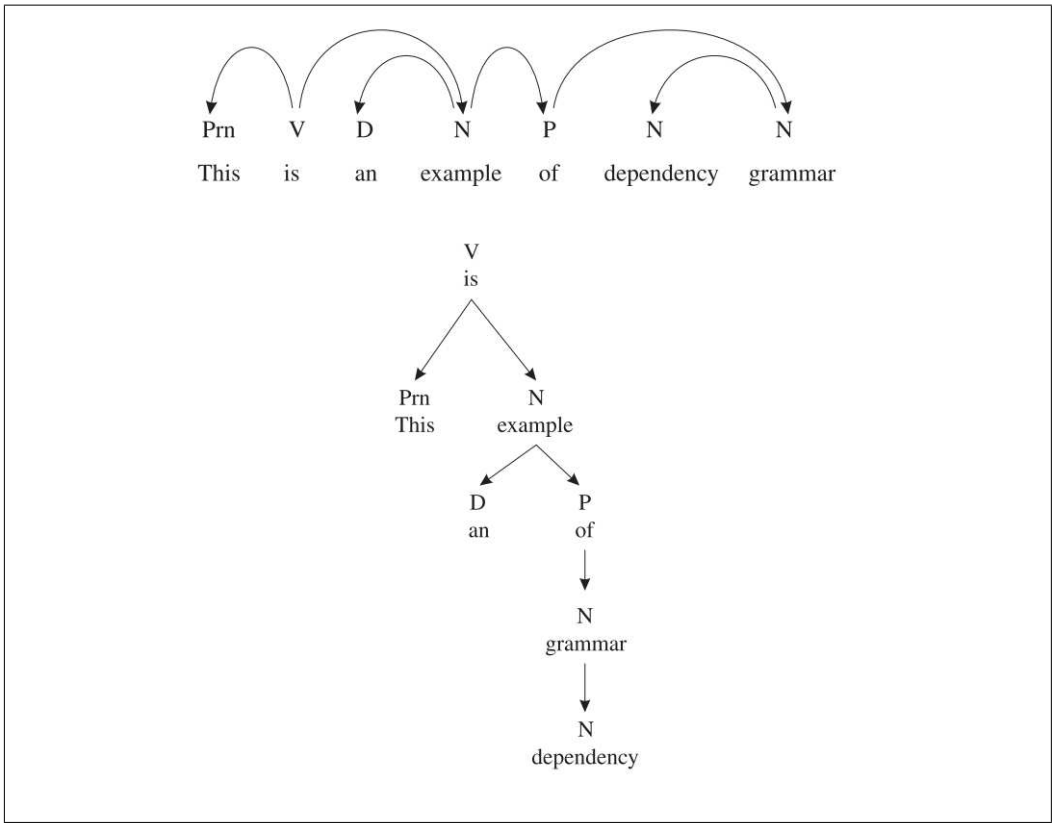

Figure 7.9: A dependency tree is a set of links connecting heads to dependents [Covington, 2001]

deep linguistic analysis which makes it difficult to use for some applications. Although automatically generating deep linguistic analysis of a given text can be accurate [Ravi et al., 2008], these high accuracies are limited to certain languages such as English. The accuracy will be lower for a text in a new domain.

In this thesis, we propose to use shallow linguistic analysis instead of deep linguistic analysis and show that if it is used with appropriate models, the results could be reasonable (see chapters 8, 9 and 10).

7.3.2 Statistical Language Analysis

Different statistical Natural Language Processing techniques were introduced in the literature [Manning and Schütze, 1999]. In this section, these techniques that are used in this thesis are explained, that is, Language Modeling and some smoothing techniques to improve Language Modeling for unseen words.

7.3.2.1 Language Modeling

Language Modeling is useful for different Natural Language Processing problems such as Machine Translation, Speech Recognition and Information Retrieval. A statistical Language Modeling assigns a probability to a sequence of words based on a probability distribution. This distribution is usually estimated from a training data. In speech recognition, this model tries to predict the next word in the speech sequence. Language Modeling is also used in Information Retrieval to calculate the probability of generating a given query from each document. In Machine Translation, Language Modeling is helpful to estimate a score for each translation based on a model built on the target language. This score helps to find the most probable sequence of words among different translation hypotheses.

Considering a sentence S as a sequence of m words, $S = w_1, w_2, ..., w_m$, the probability of having S is a multiplication of conditional probabilities. Generally, the conditional probability of having a word w_i depends on all previous words in the sentence as defined in Equation 7.14.

$$P(w_1, ..., w_m) = \prod_{i=1}^{m} P(w_i | w_1, ..., w_{i-1}) \tag{7.14}$$

In order to simplify Equation 7.14, we can use the *Markov assumption*, saying that, only the prior words (local context) affect the next word. If we consider $n-1$ prior words to estimate the current word probability, then we have an $(n-1)^{th}$ order Markov model or an n-gram model [Manning and Schütze, 1999, page 193] which is formulated in Equation 7.15.

$$P(w_1, ..., w_m) = \prod_{i=1}^{m} P(w_i | w_{i-(n-1)}, ..., w_{i-1}) \tag{7.15}$$

To estimate $P(w_i | w_{i-(n-1)}, ..., w_{i-1})$, we can simply utilize the number of times (noted $c(n-gram)$) that the n-gram $w_{i-(n-1)}...w_{i-1}w_i$ occurs in some text (corpus) divided by the number of times that $w_{i-(n-1)}...w_{i-1}$ occurs which is formulated in Equation 7.16.

$$P(w_i | w_{i-(n-1)}, ..., w_{i-1}) = \frac{c(w_{i-(n-1)}...w_{i-1}w_i)}{\sum_{w_i} c(w_{i-(n-1)}...w_{i-1}w_i)} \tag{7.16}$$

Let us follow an example to make it clear how Equation 7.15 and 7.16 can help to estimate the probability for unseen sequence of words. Consider we have a corpus of the following three sentences:

- John read Moby Dick
- Mary read a different book
- She read a book by Cher

We are going to calculate the probability of the sentence *"John read a book"* for the maximum likelihood bigram model (n=2). Considering every sentence with a tag **BOS** (Begin Of Sentence) at the beginning and **EOS** (End Of Sentence) at the end of the sentence, we have the following probabilities:

$$\begin{aligned}
p(John|BOS) &= \frac{c(BOS\ John)}{\sum_w c(BOS\ w)} = \frac{1}{3} \\
p(read|John) &= \frac{c(John\ read)}{\sum_w c(John\ w)} = \frac{1}{1} \\
p(a|read) &= \frac{c(read\ a)}{\sum_w c(read\ w)} = \frac{2}{3} \\
p(book|a) &= \frac{c(a\ book)}{\sum_w c(a\ w)} = \frac{1}{2} \\
p(EOS|book) &= \frac{c(book\ EOS)}{\sum_w c(book\ w)} = \frac{1}{2}
\end{aligned} \tag{7.17}$$

We can calculate the probability of the given sentence by using Equation 7.15 and the calculated probabilities in Equation 7.17 [Chen and Goodman, 1996] for each bigram as follows:

$$\begin{aligned}
p(John\ read\ a\ book) &= p(John|BOS) \times p(read|John) \times p(a|read) \times \\
&\quad p(book|a) \times p(EOS|book) \\
&= \frac{1}{3} \times 1 \times \frac{2}{3} \times \frac{1}{2} \times \frac{1}{2} \\
&\approx 0.06
\end{aligned} \tag{7.18}$$

Model	Parameters
1^{st} order (bigram model)	$20,000\times$ 19,999 = 400 millions
2^{nd} order (trigram model)	$20,000^2 \times 19,999 = 8$ trillions
3^{rd} order (four-gram model)	$20,000^3 \times 19,999 = 1.6 \times 10^{17}$

Table 7.5: Growth in the number of parameters for n-gram models with 20k as vocabulary size [Manning and Schütze, 1999, page 194]

In real application, it is needed to estimate n such that the n-gram can cover enough large sequence of words like *Sue swallowed the large green...* where the next word could be any of *pill, frog, etc.* But if we consider small $n = 2$, the main influent word in this sentence cannot affect the predicted word and we might have some new words including *tree, car, mountain, etc.* which are not good predictions when *swallowed* is in the sentence. But, larger n might cause other problems. For example, imagine a vocabulary of 20,000 words; an estimation for the number of n-grams would be time and memory consuming and would require a very big training corpus as listed in Table 7.5. These huge numbers of parameters for large n are not practical, so we need to reduce them. Reducing n is not the only way of decreasing the model parameters. Smaller vocabulary can also reduce the model dimensions. One way to make the vocabulary smaller is to use word stems or lemmas instead of exact words.

7.3.2.2 Smoothing Methods

In Equation 7.15, we may have some zero probabilities for unseen sequences of words which do not exist in the training corpus. These will make the equation useless; by estimating the probability of the sentence to zero. In order to overcome this problem, we can consider small probabilities for unseen words sequences, called smoothing. For example, consider the corpus in Section 7.3.2.1, when the goal is to find the probability of having *"Cher read a book"*. The probability for this sentence is zero because $p(read|Cher) = 0$ (see Equation 7.19).

$$p(read|Cher) = \frac{c(Cher\ read)}{\sum_{w} c(Cher\ w)} = \frac{0}{1} \tag{7.19}$$

Obviously, the probability of having an unseen sentence (similar to above example) cannot be zero. In these cases, smoothing techniques can help us to have better estimation for the probability of unseen events (sequences of words) by decreasing non zero probabilities and considering small probabilities for unseen events. The easiest way to have an estimation for unseen events is to pretend all events occur once more than their actual occurrences, meaning that we consider

unseen events exist once in the corpus. This is the definition of the **add-one smoothing** technique, formulated in Equation 7.20:

$$\begin{aligned} P(w_i|w_{i-(n-1)},...,w_{i-1}) &= \frac{1+c(w_{i-(n-1)}...w_{i-1}w_i)}{\sum_{w_i} 1+c(w_{i-(n-1)}...w_{i-1}w_i)} \\ &= \frac{1+c(w_{i-(n-1)}...w_{i-1}w_i)}{|V|+\sum_{w_i} c(w_{i-(n-1)}...w_{i-1}w_i)} \end{aligned} \tag{7.20}$$

where $|V|$ is the vocabulary size. Considering the previous example, the probability for *"Cher read a book"* will be 0.00003 as calculated in Equation 7.21.

$$\begin{aligned} p(\textit{Cher read a book}) &= \frac{1}{14} \times \frac{1}{12} \times \frac{3}{14} \times \frac{2}{13} \times \frac{2}{13} \\ &\approx 0.00003 \end{aligned} \tag{7.21}$$

Additive smoothing is the general form of add-one smoothing in Equation 7.20 where we consider δ instead of one for each event in the corpus (see Equation 7.22).

$$P(w_i|w_{i-(n-1)},...,w_{i-1}) = \frac{\delta + c(w_{i-(n-1)}...w_{i-1}w_i)}{\delta|V| + \sum_{w_i} c(w_{i-(n-1)}...w_{i-1}w_i)} \tag{7.22}$$

Absolute discounting [Ney et al., 1994] is another smoothing technique in which all non-zero frequent events are discounted by a small value δ and the gained frequency is uniformly distributed over the unseen events.

$$p_{abs}(w_1...w_n) = \begin{cases} \frac{(r-\delta)}{N} & \text{if } r > 0 \\ \frac{(B-N_0)\delta}{N_0 N} & \text{otherwise} \end{cases} \tag{7.23}$$

While smoothing is essential for Language Modeling, the evaluation of each smoothing technique is critical. The most common way of evaluating a smoothing technique is to evaluate the Language Modeling that uses the smoothing technique. One metric to evaluate a Language Modeling is the probability that the model assigns to a test data. We can calculate the probability of each sentence in the test data, composed of sentences $(t_1, ..., t_l)$, based on the n-gram Language Model. Then we can calculate the test set probability as the product of all sentence probabilities as defined in Equation 7.24.

$$p(T) = \prod_{i=1}^{l} p(t_i) \tag{7.24}$$

For each application based on n-gram model, we can evaluate the Language Modeling in order to find the best smoothing technique.

Term Frequency	**Definition**
Natural	$tf(t,d) = C(t,d)$, number of occurrence of term t in document d
Logarithm	$tf_l(t,d) = 1 + log(tf(t,d))$
Augmented	$tf_a(t,d) = 0.5 + \frac{0.5 * tf(t,d)}{max_t(tf(t,d))}$
Boolean	$tf_b(t,d) = \begin{cases} 1, & \text{if } tf(t,d) > 0 \\ 0, & Otherwise \end{cases}$
Log average	$tf_{la}(t,d) = \frac{1+log(tf(t,d))}{1+log(ave_{t \in d}(tf(t,d)))}$

Table 7.6: Term frequency definitions

7.3.3 Texts as Vectors and Weighting Schemes

In different applications of NLP (e.g. Information Extraction, Information Retrieval) a vectorial data representation is used. In this framework, each object (e.g. entity, relation) is represented as a feature vector. For instance, texts can be represented by vectors in which each dimension represents the weight of one word in the text. As a simple model, we can assume that all features have the same importance to all objects, so each feature value can be one for an object if it is related otherwise it is zero. Clearly, all features do not have the same importance to all objects which lead us to more sophisticated model, weighted feature vector model [Salton and Buckley, 1988]. In this model each feature is weighted based on its importance to the object and its condition. Among different weighting scheme in the literature, we explain tf-idf as it is popular in different application.

The **tf-idf** (Term Frequency - Inverse Document Frequency) is used to weight a term in a document (of a corpus) based on its importance to that document. Considering the term weighting in a collection of documents (a corpus), we expect to have higher weight for a term in a document if it is not frequent in majority of the documents. Similarly, we expect to have lower weight for a term which is, almost equally, distributed among different documents. The tf-idf is defined based on the term frequency and inverted document frequency. Different definitions are used in the literature for term frequency [Manning et al., 2008] which are listed in Table 7.6. Each of these definitions can be used for the tf part of the tf-idf definition in Equation 7.26.

Inverse document frequency is a measure to find out whether a term is popular among documents or not. It is defined by the logarithm of the division of the total number of documents ($|D|$) by the number of documents containing the

term (Equation 7.25).

$$\text{idf}\ (t, D) = log(\frac{|D|}{|\{d \in D : t \in d\}|}) \tag{7.25}$$

Finally, tf-idf for term t in document d of collection D is defined as the multiplication of tf and idf in Equation 7.26

$$\text{tf-idf}\ (t, d, D) = \text{tf}\ (t, d) * \text{idf}\ (t, D) \tag{7.26}$$

Tf-idf is the most widely used weighting scheme in the literature, but there are some drawbacks which make it difficult to be used for some applications. First of all, the complexity of the tf-idf is $O(n^2)$ (where n is the number of documents in the corpus) which could be a problem for large data or live stream of documents such as the Web. Reed et al. [2006] proposed a new term weighting scheme based on the inverse corpus frequency with a better complexity of $O(n)$. They argue that we can estimate the document frequency distribution for larger data based on the smaller data set. Then, a pre-calculated distribution can be used. In this way, generating a weighted document vector is as simple as a table lookup. Moreover, in this way, we do not need to have the whole corpus in advance to calculate the weights. So, it is more practical for a dynamic corpus such as the Web.

The tf-idf is an unsupervised weighting scheme since it does not use any information from the training data. Different supervised weighting schemes have been proposed in the literature [Debole and Sebastiani, 2003; Soucy, 2005]. Soucy [2005] proposed a new supervised weighting scheme which can use training data to improve the idf score. To highlight the importance of supervised weighting schemes, consider the application of detecting text language. This is a problem of text clustering in which common terms, such as *the* in English or *les* in French, can play the main role in the language detection. However, the idf score for these terms is very low due to the high frequency of the term in the corpus.

In this thesis, we use a modified version of tf-idf in Chapters 9 and 10.

7.4 Conclusion

In this chapter, we reviewed some technical backgrounds about Information Extraction, Machine Learning and Natural Language Processing. In each section, we only covered a limited number of techniques that are used in this thesis, either in our proposed models or as state-of-the-art techniques. We explained some basics about Information Extraction and its applications. Different kinds of information that can be extracted were also discussed. In this thesis, we use different Machine Learning algorithms belonging to the two main groups that we presented:

supervised and unsupervised models. In Section 7.2, we discussed some popular algorithm in both categories. We covered two algorithms, k-Nearest Neighbors and Support Vector Machine as two supervised algorithms that will be exploited in the work presented in the next chapters. Then, some clustering algorithms (i.e. k-means and Markov Clustering Algorithm) are reviewed as unsupervised techniques. Finally, different evaluation metrics were also discussed. Since, in this research we deal with textual data, some techniques are studied in Section 7.3 to process text. We reviewed these techniques in two main groups: linguistic and statistical analysis.. In the linguistic analysis sub-section, some deep and shallow analysis techniques (e.g. POS tagging, syntactic parsing) are reviewed. Then, we explained Language Modeling and some smoothing approaches as statistical analysis techniques. We also explained some weighting schemes for vectorial data representation in this section.

In this thesis, we aim to propose modules for Information Extraction; besides the good performance, we want them to be simple in order to be, eventually, used in environment challenging their robustness. Based on our explanations in Section 7.1.2 about different kinds of extractable information, we proposed two models for relations and one model for entities in this thesis. Their representation in this manuscript is organized as follows. In the next chapter, we propose a supervised model to extract relations among entities in a sentence. In this model, we use n-gram model to build the feature vector for each relation. We propose a similarity function based on Language Modeling which is used with a kNN algorithm to classify relations. Then, in Chapter 9, we move to unsupervised model to discover relations among entities in a sentence. Here, we proposed a new similarity function based on Language Modeling and average probability. We use the proposed function with Markov Clustering algorithm to cluster relations. Since all relations are defined between entities in text, we define our last research topic as discovering entities. In Chapter 10, we propose a model to cluster entities (more precisely, proper nouns) semantically in text. To do so, we use a new data representation called Bag-of-Vectors instead of popular Bag-of-Words model. Moreover, to define contextual information for each entity, we utilize n-gram of words instead of words in order to capture the words sequences. To calculate similarity between entities with the Bag-of-Vectors, we adapt classical similarity functions. In addition, we define a discriminative similarity function to define the similarity between entity pairs according to their most important contextual information. Finally, these chapters are followed by some discussions and conclusions in the last chapter.

Chapter 8

Relation Extraction

8.1 Introduction

Since the nineties, a lot of research work has been dedicated to the problem of corpus-based knowledge acquisition, where the aimed knowledge is terminology, special cases of vocabulary (e.g. named entities), lexical relations between words or more functional ones. The research presented in this chapter focuses on this last kind of acquisition, i.e., Relation Extraction. Regarding to the main goal of this thesis toward a robust Information Extraction system, Relation Extraction is defined as a research task in this research. We define this task as semantic annotation of relations between objects (entities) in textual data. In this chapter, we propose a supervised model to extract and classify relations.

In order to evaluate the proposed model, we experiment on real data with some common properties to transcribed text. We assume that transcribe texts are noisy, so we cannot use deep linguistic analysis techniques. Besides we aim to work with relatively small data set since, in the final system, the transcriptions of multimedia documents are small. First, we need to choose a similar and challenging problem. Among all Relation Extraction data, we evaluate our models on a challenging task in bio-medical texts. We apply the proposed model on Protein-Protein Interaction (PPI) extraction. The complexity of sentence in the selected data is similar to our ideal data (transcribed text).

The goal of PPI is to find pairs of proteins within sentences such that one protein is described as regulating, inhibiting, or binding the other. In functional genomics, these interactions, which are not available in structured database but scattered in scientific papers, are central to determine the function of the genes. In order to extract PPIs, the texts which contain the interactions have to be analyzed. Two kinds of linguistic analysis can be used for this purpose: deep and shallow which are explained in Section 7.3.1. Automatic deep analysis, which provides a syntactic or semantic parsing of each sentence, can be a useful source

of information. However, tools for automatic deep analysis are available only for a limited number of natural languages, and produce imperfect results. Manual deep analysis, on the other hand, is time consuming and expensive. Another way to analyze texts is to rely only on a shallow linguistic analysis, taking into account the sole words, lemmas or parts of speech (POS) tags. Automatic tools for shallow analysis are available for many languages, and are (sufficiently) reliable.

In this chapter, we advocate the use of shallow linguistic features for Relation Extraction tasks. First, we show that these easy and reliable pieces of information can be efficiently used as features in a machine learning (ML) framework, resulting in good PPI extraction systems, as effective as many systems relying on deep linguistic analysis. Furthering this idea, we propose a new simple yet original system, called LM-kNN based on language modeling, that out-performs the state-of-the-art systems.

The chapter is organized as follows. Section 8.2 reviews related work on Relation Extraction in general and for bio-medical texts. Section 8.3 specifies the problem and our methodology. Results when using classical machine learning algorithms are given in Section 8.4, together with a comparison with existing systems. Some conclusions are provided in last section.

8.2 Related Work

Relation Extraction is a task of extracting predefined relations between entities in texts (or sentences) by learning from labeled data. Recognizing entities is defined as a Named Entities Recognition task where state-of-the-art systems can automatically annotate data with high accuracy, near human performance [Marsh and Perzanowski, 1998]. Relations in text could be binary, for example *located-in(CMU, Pittsburgh)*, or higher order relations as well (see Section 7.1.2 for the example of disease outbreak).

In order to extract relations, the texts which contain the relations have to be analyzed. Two kinds of linguistic analysis can be performed for this purpose: deep and shallow (see Section 7.3.1). We review different models of Relation Extraction in two groups based on how much deep analysis is used in the model.

As explained in Section 8.1, we provided the experimental results of Relation Extraction on a PPI extraction task. This literature review focus is set on researches dedicated to Relation Extraction in general then, for bio-medical texts, especially those evaluated in a Protein-Protein Interaction (PPI) extraction framework.

The systems proposed for this task can be organized into different groups, depending on the source of knowledge (deep vs. shallow linguistic information) and on the approach used (manual vs. ML).

As an instance of rule-based Relation Extraction systems, RelEx [Fundel et al., 2007] exploits manually built extraction rules handling deep and shallow linguistic information. First, a chunk dependency parse tree is built for each sentence. Then, the authors proposed three rules to create candidate relations by extracting paths connecting protein pairs from the dependency parse tree. These tree rules, considered as frequently used rules in English to describe relations, are listed below:

1. effector-relation-effectee (A activates B)
2. relation-of-effectee-by-effector (Activation of A by B)
3. relation-between-effector-and-effectee (Interaction between A and B)

In addition to these above rules, some filters are defined to remove false relations which can be generated by the rules. For example, negated relations are excluded from the candidates by using a list for negation words such as *no*, *not*, *without*, *lack*, *unable*, etc. Additionally, a list of *relation restriction terms* including 1048 restrictions is used to remove relations that do not have at least one of these terms in the path between proteins pair. This system yields good results; yet using such a hand-elaborated knowledge is a bottleneck requiring expertise for any new domain.

Thus, many ML-based approaches were proposed to overcome this limitation. The ML techniques range from SVM with complex kernels [Airola et al., 2008; Kim et al., 2010] or CRF [Li et al., 2007], to expressive techniques like inductive logic programming [Phuong et al., 2003]. In the following, we review some of these models in two main groups based on how deep linguistic information they needed.

Linguistic information, in the simplest way, is a set of words in a text (or a sentence) or is a semantic or syntactic information (see Section 7.3.1 for more details about deep and shallow linguistic information).

State-of-the-art Relation Extraction systems use syntactic analysis of a sentence to extract relation between entity pairs. These models employ Support Vector Machine and kernels to classify relations. Culotta and Sorensen [2004] proposed different kernels based on the smallest common sub-tree between two entities (including them) in an augmented dependency tree of the sentence. This short sub-trees helped them to reduce noises in the training data which take into the account local characteristics of relations. For each node in the dependency tree, they assigned different levels of features (e.g. word, POS, entity-type, hypernyms) to build different kernels. Finally a SVM is used for classifying relations. The evaluation showed better precision compared to a simple Bag-of-Words model; but the recall was low. The authors argued that the main reason for the

low recall it the high frequency of heterogeneous negative relations. Following up the use of smallest sub-tree in the parse tree, Bunescu and Mooney [2005] proposed a model, called shortest path, to improve the performance of a Relation Extraction tasked compared to Culotta and Sorensen [2004]. The shortest path hypothesis says that if there is a relation between two entities in a sentence, it is concerned to the shortest path between entities in a undirected dependency graph. A kernel is defined based on this hypothesis in order to calculate the similarity between two relations. Different features are utilized in this computation including the word, POS, and the word semantic class (e.g. person, location). Experimental results showed performance improvement compared to Culotta and Sorensen [2004]; but the recall was still low. In Bunescu and Mooney [2005] model, the similarity between two shortest paths (relations) is high only if they have the same length. In order to increase the recall, Choi et al. [2012] proposed a 3-gram model to calculate the similarity between two sub-trees. In this model, each 3-grams of two sub-trees are considered to calculate the similarity instead of the whole sub-trees as proposed in Culotta and Sorensen [2004] and Bunescu and Mooney [2005] models. Experimental results on Relation Extraction from news articles show higher recall so f-measure while the precision is near to the state-of-the-art system.

Using syntactical information is not always possible either because of having noisy data or the preparation cost of such information. In these cases, using shallow information is a solution as we explained before in Chapter 7.3.1. Lexical or linguistic features of words surrounding a pair of entities can be considered as shallow linguistic features to train the systems [Bunescu and Mooney, 2006; Giuliano et al., 2006; Sun et al., 2007]. Bunescu and Mooney [2006] defined subsequence kernels based on 4 patterns including words before-between, between, between-after pair of entities with the 4th patterns which indicated that there is no word between entities (see Figure 8.1). In addition to words in these 4 patterns, they also use word class such as POS and chunk head as features in their model. Considering each set of features as a separate feature space (set of words, POS, etc.), they defined a relation kernel based on string kernel [Lodhi et al., 2002] to calculate the number of common patterns between two sentences which is a sum of three different subsequence kernels. The relation kernel is used in conjunction with SVM learning in order to classify the relation as negative and positive.

Some of the state-of-the-art models are also used in a more specific task of Relation Extraction: Protein-Protein Interaction extraction. Xiao et al. [2005] investigate how different shallow linguistic information can affect the Protein-Protein Interaction extraction. Different shallow linguistic features, used in this research, are listed below:

- **Words**: It includes protein names with words between and surrounding

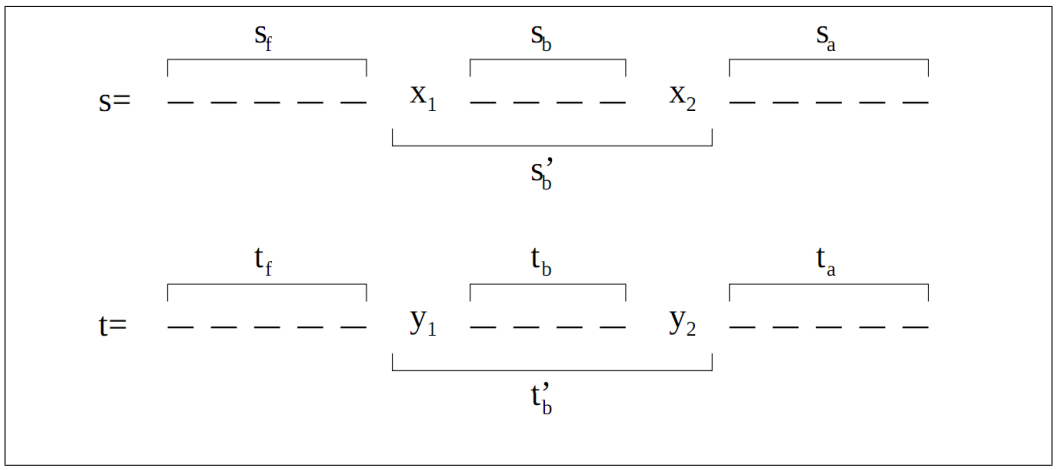

Figure 8.1: Sentence segments

protein pairs as a bag-of-words.

- **Overlap**: The number of other protein names that exist between proteins pair is considered as a feature.
- **Keyword**: If there is any keyword between or around protein pairs (based on a predefined list of keywords), these keywords with their positions are considered as features.
- **Chunks**: Heads of chunks between and around proteins pair are considered as a feature without considering their positions. In addition, all chunk types (e.g. verb, noun) are also included in this feature set.
- **Parse tree**: The shortest path (between proteins pairs) extracted from the parse tree of the sentences is considered as a feature.
- **Dependency tree**: It has two features, the first one as a boolean flag if there is any connection between the protein pair in the parse tree. The second feature is the word/POS of the common root between the protein pairs.
- **Head of proteins pair**: A combination of two protein name heads is considered as a feature (e.g. *prion_kinas* for *"bovine prion"* and *"protein kinas"* as a protein pair).
- **Abbreviation pairs**: For each protein name, they considered an abbreviation in which the pair of abbreviation makes another feature (e.g. *bprp_protein_kinase* for above pair).

Different combination of above features are used to extraction PPI from a corpus. Among all, only some of them are useful for this task. Words between and surrounding proteins pairs, their heads and pairs of abbreviations show more performance improvement in terms of precision and recall. Other features, especially the parse and the dependency trees not only were not useful to improve the performance but also decreased the performance due to the error in the parsing phase.

So far, all researches in Relation and PPI Extraction were about how to define features. Kim et al. [2010] proposed to shift the focus of bio-medical Relation (e.g. PPI) Extraction, from the problem of pattern extraction to the problem of kernel construction for using SVM as classification tools. So they proposed four kernels including predicate, walk, dependency and hybrid kernels which are considered in two main groups, feature-based kernels and structure-based kernels. Kernel functions in the feature-based kernels are similar and the only difference is about how to build the feature vectors; for example, a feature vector based on *(word1, relation, word2)* and another one based on *POS1, relation, POS2)*. On the contrary, structure-based kernels are different; they can provide a formalism to learn the structured data such as graph or tree. A **predicate kernel** is built assuming that a predicate and its arguments are important for Relation Extraction. That is, a pair of entities is related if an interaction predicate exists in the shortest path between them and both entities are arguments of that predicate. In the **walk kernel**, the structural information is defined by walks in a graph based on the parse tree of the sentence. Kim et al. defined two kinds of walk kernels, called v-walk and e-walk. V-walk is a path of vertices or edges beginning from one vertex and end at another vertex. Besides v-walk, e-walk is defined that starts and ends with an edge. These two walk kernels provide a syntactic structure for the learning model. A lexical walk and syntactical walk are defined for both of v-walk and e-walk . Lexical words and dependency relations build lexical walk and syntactical walk is defined by POS and dependency relations. Walk kernels can consider more detailed linguistic information compared to the predicate kernels since it takes into account all information in the shortest path between entity pairs.

Predicate and walk kernels as two feature-based kernel can fail to identify similar relations due to their sensitivity to small changes in parse trees. Moreover, structured data such as trees and graphs cannot be represented by flat features. So, structure-based kernels were introduced; the similarity in this kernels is directly calculated between two trees or graphs by investigating common sub-graphs. Based on this definition, **dependency** and **hybrid** kernels were proposed. The similarity between two nodes in a graph is the number of common words and their POS dependency sub-graph (shortest path in the parse tree) between the protein names. So dependency kernel is a word-level model based

on the words and their POSs in the dependency graph. The hybrid kernel is a combination of dependency kernel with walk kernel in order to cover the similar path between two protein names. These structure-based kernels have still problem with non-contiguous sub-graph and partial match between two nodes [Kim et al., 2010]. Moreover, a walk-weighted subsequence kernel was proposed to parametrize non-contiguous syntactic structure as well as semantic roles and lexical features. This new kernel can improve the learning from small amount of training data.

Zhou et al. [2010] proposed a model to enrich shortest path trees with semantic information as well as contextual information in order to cover more complex relations. In the shortest path tree features, the relation normally depends on the words or POS tags between entity pairs. But for example for the sentence *"John and Mary got married"*, obviously, words or shortest path tree between John and Mary cannot help to classify the relation between these entities. In the proposed model, the authors enriched the shortest path tree with contextual information around entity pairs and refine the tree by removing unnecessary components and compressing coordinated conjunction into a single node using some heuristic rules. These shortest path tree enrichment with semantic expansion together with a context-sensitive convolution tree kernel helped them to outperform state-of-the-art models.

As one can see, different deep and shallow linguistic features are used for PPI extraction as described in previous paragraphs. Sætre et al. [2008] investigate the effect of the combination of these different features. First, lemmatized words as Bag-Of-Words (BOWs) are used as features to build a baseline system. In this model, they consider words before, between and after protein pairs as a BOW. They also tried to replace BOWs by features coming from a single parser, HPSG (Head-Driven Phrase Structure Grammar) [Pollard and Sag, 1994] or dependency, but it did not improve the results. However, when they combined features from the two parsers, it improved the results. They also argued that, in the combined model, while a fast dependency parser can improve the base-line system results, the HPSG parser, as an expensive parser, can contribute to a bigger improvement compared to the dependency parser. Finally, the best results are obtained when a combination of dependency and HPSG parser features are used with word features (BOWs) such as word lemmas and POS tags.

Lexical variety in some domains such as bio-medical can be a challenge, especially when the training data is relatively small. Fayruzov et al. [2008a] proposed a model to rely on more general language structures such as parsing and dependency information to build the feature vector. For any supervised Relation Extraction, a training data is an essential requirement. The quality of the training data is important as its quantity. A supervised model that is trained on larger annotated data generally shows better results compared to a trained model with

less data. For example, Miyao et al. [2009] show that using larger training data to train a parser can improve the parser accuracy which improves the Protein-Protein-Interaction extraction as a consequence. Although the quantity of the training data is not the only parameter. The quality of the training data is very important. It is reported [Miyao et al., 2009] that improving 1% of the parser outputs can improve 0.25% of the performance of Protein-Protein-Interaction (PPI) extraction. Moreover, moving from manually built dependency trees to automatically generated tree can drop the performance of a PPI extraction by 15% [Ebadat, 2011].

Indeed, grammatical relations are assumed to be important for relation extraction, especially when few training data compared to test data are available [Fayruzov et al., 2009]. Yet, the performance of extraction systems being sensitive to the accuracy of automatic parsers [Fayruzov et al., 2008b], shallow linguistic information still remains an option [Xiao et al., 2005], though up-to-now less effective than deep one. Furthermore, bio-medical texts often contain complex sentences with long range relation which cause parsers generate wrong parse tree.

In this work, we defend the hypothesis that shallow linguistic information combined with standard ML approaches is sufficient to reach good extraction results. Furthermore, we propose a system demonstrating that when this simple information is cleverly used, it even out-performs these state-of-the-art systems.

8.3 An Instance-based Learning Model

This section is dedicated to the different machine learning approaches, based on shallow linguistic features, that we experimented to extract relations from texts. The two first subsections respectively present how to model the task as a machine learning problem —and in particular how relations are described— and the classification tools commonly used for similar tasks. In the last subsection, we propose a new Relation Extraction technique, based on language modeling, which is expected to be more efficient than the existing ones.

8.3.1 Problem Analysis

The goal of Relation Extraction is to predict, at the occurrence level, if two entities share a defined relation. Expert systems, with manually defined extraction patterns, are usually very costly to build, cannot be adapted to new domains and require an expert knowledge both for the pattern design and the domain which is rarely available. Thus, it is usual to try to build Relation Extraction systems by machine learning. Such approaches require examples of the spotted relations, but the necessary expert knowledge is cheaper in this case than for pattern de-

sign. Moreover, bootstrapping and iterative approaches [Hearst, 1992] or active learning can be used to lower this cost.

In PPI extraction that we adapt for our experiments, the goal is to predict if there is any interaction between two proteins. In such a case, the relation is directed, that is, one of the entity is an agent and the other is the target. For example, in the sentence reported in Figure 8.2 for bold entities (proteins), there is a relation between *GerE* and *cotD* for which *GerE* is the agent and *cotD* is the target.

> ***GerE*** *stimulates* ***cotD*** *transcription and inhibits* ***cotA*** *transcription in vitro by* ***sigma K*** *RNA polymersase, as expected from in vivo studies, and, unexpectedly, profoundly inhibits in vitro transcription of the gene (****sigK****) that encode* ***sigma K****.*

Figure 8.2: Sample sentence for protein-protein interaction

To handle this directed relation problem, we model it as a 3-class machine learning task. For each training sentence, each pair of entities is either tagged as *None* if the entity pair does not have any interaction, *LTR* if the interaction is from the left to the right (agent to target in the sentence word order), and *RTL* if the interaction is from the right to the left.

8.3.2 Bag of Lemmas Data Representation and Machine Learning

The representation, that is, the set of features describing our examples for the machine learning algorithms is voluntarily chosen as very simple. Indeed, a relation is simply represented by the bag of lemmas occurring between the two entities. Grammatical words are kept since they may be important clues to detect the direction of the interaction (like the word *by*). For instance, Table 8.1 reports the examples found in the sentence: *Most cot genes, and the gerE, are transcribed by sigma K RNA polymerase.* More formally, each example is described by a vector; each dimension of this vector corresponds to a lemma and its value is 1 if the word occurs between the entities and 0 otherwise. The sparse vector obtained is expected to be a representation both performant and robust since it does not rely on any complex pre-processing.

In the experiments reported below, this bag-of-lemmas representation is exploited with machine learning techniques popularly used for similar tasks: Support Vector Machine (SVM), Random Tree and Random Forest (as implemented in the Weka toolkit [Hall et al., 2009]).

Example pair	Bag of lemmas	Class
cot,gerE	gene,and,the	None
cot,sigmaK	gene,and,the,gerE, gene,be,transcribe,by	RTL
gerE,sigmaK	gene,be,transcribe,by	RTL

Table 8.1: Examples of bags of lemmas to be used as feature vectors

SVM aims at constructing a set of hyperplanes in the representation space dividing the examples according to their class. When used with complex kernels, the hyperplanes are searched in a higher space, resulting in a complex separation in the original representation space (see Section 7.2.1.2 for more details). Random Tree and Random Forest [Breiman, 2001] are two classification algorithms based on the well-known decision trees offering a better robustness especially when tackling problems with small or noisy training data. Random Tree constructs a classical decision tree but considers only a subset of attributes (features) that are randomly selected at each node. Random Forest extends this technique: it builds a large set of decision trees by randomly sampling the training data and the features. It is important to note that all these techniques learn explicitly or implicitly to divide the representation space—in our case the lemma vector space—into different parts corresponding to our 3 classes.

8.3.3 Nearest Neighbors with Language Modeling

Besides these somewhat classical machine learning approaches, we propose a new technique to extract relations. As the previous ones, it still uses shallow linguistic information, which is easy to obtain and ensures the necessary robustness. One of the main differences with the previous approaches concerns the representation of the examples: it takes into account the sequential aspect of the task with the help of n-gram language models. Thus, a relation is represented by the sequence of lemmas occurring between the agent and the target, if the agent occurs before the target, or between the target and the agent otherwise. A language model is built for each example Ex, that is, the probabilities based on the occurrences of n-grams in Ex are computed; this language model is written $\mathcal{M}_{Ex}$. The class (LTR, RTL or none) of each example is also memorized.

Given a relation candidate (that is, two proteins or genes in a sentence), it is possible to evaluate its proximity with any example, or more precisely the probability that this example has generated the candidate. Let us note $C = \{w_1, w_2, ..., w_m\}$ the sequence of lemmas between the proteins. For n-grams of n

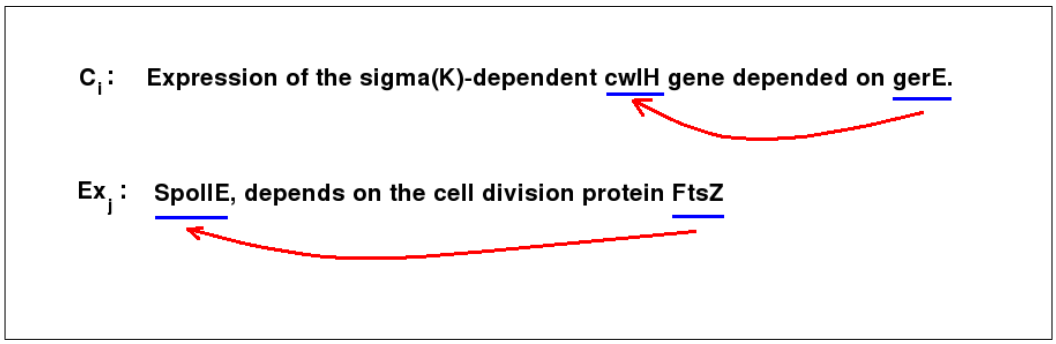

Figure 8.3: Example of two sentences from bio-medical domain

lemmas, this probability is classically computed based on Equation 8.1.

$$P(C|\mathcal{M}_{Ex}) = \prod_{i=1}^{m} P(w_i|w_{i-n}..w_{i-1}, \mathcal{M}_{Ex}) \tag{8.1}$$

As an example of computing this probability, consider two sentences in Figure 8.3 in which C_i is the i^{th} candidate and Ex_j is j^{th} example. We estimate all conditional probabilities for 2-grams between pair of entities in Ex_j based on Language Modeling as explained in Equation 7.16. These probabilities are considered as $\mathcal{M}_{Ex_j}$ and are used to estimate the probability of having 2-grams in C_i which is $P(C_i|\mathcal{M}_{Ex_j})$ as explained in Equation 8.2.

$$P(C_i|\mathcal{M}_{Ex_j}) = P(gene|BEGIN, \mathcal{M}_{Ex_j}) * P(depend|gene, \mathcal{M}_{Ex_j}) * P(on|depend, \mathcal{M}_{Ex_j}) \tag{8.2}$$

As for any language model in practice, probabilities are smoothed in order to prevent unseen n-grams to yield 0 for the whole sequence. For example, in Equation 8.2, the probability of having "*gene depend*" is zero based on the probabilities estimated on Ex_j. In the experiments reported below, we consider bigrams of lemmas and simply use interpolation with lower order n-grams (unigram in this case) combined with an absolute discounting [Ney et al., 1994].

The probability defined in Equation 8.1 can not be used as similarity function since it is not symmetric function (see properties of similarity functions in Section 7.2.2.4). In addition, in order to prevent examples with long sequences to be favored (being similar to many candidates), the probability of generating the example from the candidate ($P(Ex|\mathcal{M}_C)$) is also taken into account. Finally, the similarity between an example and a candidate is defined as the minimum of these two probabilities in Equation 8.3.

$$sim(Ex, C) = \min\left(P(Ex|\mathcal{M}_C), P(C|\mathcal{M}_{Ex})\right) \tag{8.3}$$

The class is finally attributed to the candidate by a k-nearest neighbor algorithm: the 10 most similar examples (highest sim) are calculated and a majority vote is performed. This lazy-learning technique is expected to be more suited to this kind of tasks than the model-based ones proposed in the previous sub-section since it better takes into account the variety of ways to express a relation (see Section 8.4.3 for a discussion on this issue). In the rest of the text, this technique is called LM-kNN.

8.4 Experiments

In this section, the experiments with the different relation extraction systems described above are presented. The data used and the evaluation metrics and methodologies are first detailed. Then the results obtained through cross-validation and on held-out test data are given and compared with existing systems. Finally, some insights raised by these results are provided.

8.4.1 LLL Data

To evaluate the different proposed Relation Extraction systems, we use the data developed for the Learning Language in Logic 2005 (LLL05) shared task [Nédellec, 2005]. The goal of LLL05 was to extract protein/gene interactions in abstracts from the Medline bibliography database.

There are different reasons to explain why we use these data for evaluation instead of transcription of multimedia documents as it is aimed in the thesis (see also explanations in Chapter 6). With these chosen data, we have different evaluation opportunities: In addition to the cross-validation, we can evaluate our method on unseen data, provided by the shared task LLL05. We also have opportunities to compare our proposed model to other groups in Quaero (see more information in Chapter 6). In addition, we assumed that it shares some slight properties with textual data related to multimedia documents in term of having frequent unknown words. In this point of view, the evaluation results computed here could be similar for multimedia.

An example of LLL05 data is reported in Table 8.2. In LLL05, all sentences are tokenized and interactions between entities are annotated.

Table 8.2 shows an example of basic data provided by LLL05. In addition, they prepared a deeper linguistic information of sentences as another training data set. An example of these data are reported in Table 8.3.

The provided training set is composed of 80 sentences in which a total of 271 interactions between genes/proteins are identified. These sentences are provided in two groups, without co-references and with co-references which are detailed in Table 8.4.

Data type	Example
sentence	ykuD was transcribed by SigK RNA polymerase from T4 of sporulation.
words	word(0,'ykuD',0,3) word(1,'was',5,7) word(2,'transcribed',9,19) word(3,'by',21,22) word(4,'SigK',24,27) word(5,'RNA',29,31) word(6,'polymerase',33,42) word(7,'from',44,47) word(8,'T4',49,50) word(9,'of',52,53) word(10,'sporulation',55,65)
agents	agent(4) : SigK
targets	target(0) : ykuD
genic interactions	genic_interaction(4,0) : ykud ← SigK

Table 8.2: Example of LLL05 basic data

Data type	Example
sentence	Localization of SpoIIE was shown to be dependent on the essential cell division protein FtsZ.
words	word(0,'Localization',0,11) word(1,'of',13,14) word(2,'SpoIIE',16,21)
lemmas	lemma(0,'localization') lemma(1,'of') lemma(2,'SpoIIE')
syntactic relations	relation('comp_of:N-N',0,2) relation('mod_att:NADJ',13,10) relation('mod_pred:N-ADJ',0,7) relation('mod_att:N-N',14,13)
agents	agent(14)
targets	target(2)
genic interactions	genic_interaction(14,2)

Table 8.3: Example of LLL05 enriched data

Name	Sentences	Interactions
Without co-references	57	106
With co-references	23	165
Total	80	271

Table 8.4: LLL05 statistics

Since only positive examples (RTL or LTR in our case) are provided in the training data, we need to consider negative examples for training. As explained before, all interactions are directed; thus, each protein pair within a sentence having no interaction between its constituents is considered as a negative example. The test set is composed of another set of sentences for which the ground-truth is kept unknown; the results are computed by submitting the predictions to a web service. The original LLL challenge offered the possibility to train and test the systems only on interactions expressed without the help of co-references (mostly with pronouns designating a previously mentioned entity). The training and test data were also provided with or without manual syntactic annotations of the sentences (dependency analysis). Of course, in order to evaluate our systems in a realistic way, we used the data containing interactions expressed with or without co-references, and we did not considered the manual syntactic annotation.

8.4.2 Evaluation

The evaluation metrics chosen in our experiments are those classically used in this domain: precision, recall and f-measure. It is important to note that in this evaluation, partially correct answers, like an interaction between two entities correctly detected but with the wrong interaction direction, are considered as wrong answers.

We evaluate our LM approach and compare it with the more traditional machine learning techniques and the state-of-the-art systems in two ways. First, we classically use cross-validation. Yet, with so few examples, it is important to choose a number of folds important enough to provide reliable figures; in the results presented below, 30-fold cross-validation is considered. The second way is by using the unseen test dataset. This dataset was developed for the evaluation of the LLL challenge. As mentioned above, the ground-truth is kept unknown; and the results are computed by submitting the predictions to a web service.

The differences between these two evaluation procedures shed light on inherent difficulties and biases in some studies that we discuss after presenting our results.

8.4.2.1 Cross-Validation Evaluation

Table 8.5 reports the recall (R), precision (P) and f-measure (F) computed by 30-fold cross-validation on the complete training data (with and without co-references) provided by LLL05 (Table 8.4). The results are obtained with different machine learning techniques presented in the previous section. More precisely, the SVM used is the popular libSVM implementation [Chang and Lin, 2001], which was tested with usual kernels (linear and RBF); we tested with different values of γ and only display those obtaining the most interesting results. Random Forest

was used with 700 trees, and Naive Bayes and Random tree were used with their default parameters in Weka if any.

Algorithm	**P**	**R**	**F**
libSVM linear kernel	77.1	77.4	77.2
libSVM RBF kernel ($\gamma = 0.1$)	40.7	63.8	49.7
libSVM RBF kernel ($\gamma = 0.5$)	81.4	74.9	78
Random Forest	80.4	80.6	80.4
Random Tree	77.6	77.4	77.5
Naive Bayes	75.1	68.1	69.3
Naive Bayes Multinomial	70.4	70.3	70.3
LM-kNN	**82.2**	**80.3**	**81.2**

Table 8.5: Performance of shallow linguistic based techniques with 30-fold cross-validation

It is interesting to note that all the techniques perform well with our representation, achieving reasonable score, except for the SVM with a RBF kernel and $\gamma = 0.1$. This negative result can be explained by the fact that the SVM with such settings and so few training data has a tendency to over-fit, especially because of the training data amount. Apart from this problem, the closeness of the other results tends to show that, for the same bag-of-lemmas representation, the choice of the classifier does not strongly impact on the performance. Yet, overall, Random Forest, SVM with adequate settings and our LM-kNN technique show the highest f-measures.

8.4.2.2 Held Out Data Evaluation

The held out test data provided for the LLL challenge allows us to evaluate the previous techniques in another evaluation framework. Table 8.6 reports the performance obtained by these techniques on the complete test set (interaction expressed with or without co-references). For comparison purposes, the results on this dataset reported by other studies are also included. Since many teams have only considered the evaluation without coreferences, which is supposed to correspond to an easier task, we also report the results of our LM-kNN approach and other state-of-the-art systems in this context in Table 8.7. The first part of each table concerns systems using raw data (no manual annotation), which

corresponds to a realistic evaluation of the systems, and the second part contains results of other systems using the provided manual syntactic analysis.

System	**P**	**R**	**F**
systems on raw data			
Goadrich et al. [2005]	25.0	81.4	38.2
Random Forest	57.9	48.1	52.6
libSVM linear kernel	58.0	56.6	57.3
LM-kNN	**70.9**	**79.5**	**75**
systems on manually annotated data			
Katrenko et al. [2005]	**51.8**	**16.8**	**25.4**
Goadrich et al. [2005]	14.0	93.1	24.4

Table 8.6: Results for held-out test set of LLL, with or without co-references

System	**P**	**R**	**F**
systems on raw data			
Hakenberg et al. [2005]	50.0	53.8	51.8
Greenwood et al. [2005]	10.6	98.1	19.1
Kim et al. [2010]	68.5	68.5	68.5
Fundel et al. [2007]	68	78	72
LM-kNN	**67.1**	**87**	**75.8**
systems on manually annotated data			
Popelínský and Blaťák [2005]	37.9	55.5	45.1
Riedel and Klein [2005]	60.9	46.2	52.6
Kim et al. [2010]	**79.3**	**85.1**	**82.1**

Table 8.7: Results for held-out test set of LLL, without co-references

The first thing one can note from Table 8.6 is that the results are lower than those obtained by cross-validation. This loss is particularly important for the classical approaches based on a bag-of-lemmas representation. This point is not specific to our approaches and was already noticed by previous studies using the LLL dataset. It is due in part to a difference between the way the training and the test sets were built: the distributions of positive examples and negative ones are very different in these two sets since the test data contains much more sentences without any valid interaction. With respect to this, our LM-kNN approach over-performs the other ones and still produces high results for this task.

Besides our LM-kNN technique which ranks first (+6.5% over the best known results for fully automatic systems), it is interesting to note that our other machine learning approaches also perform well compared with state-of-the-art techniques, even though the latter could be considered as more complex than our methods. Indeed, Hakenberg et al. [2005] used finite state automata to generate extraction patterns. In addition to LLL corpora, these authors took advantage of 256 additional positive examples manually annotated. The method of Greenwood et al. [2005] generates candidate patterns from examples with the help of MiniPar for a syntactic analysis and WordNet and PASBio for a semantic analysis and tagging. Goadrich et al. [2005] applied Inductive Logic Programming and Markov Logic methods. The approach used by Kim et al. [2010], as we explained in Section 8.2, relies on the shortest path in the syntactic parse tree and a specially developed kernel for SVM.

Results of systems tested with manual syntactic information are also worth noting. Katrenko et al. [2005] used the manual syntactic annotations and a ad'hoc ontology to induce extraction patterns. Popelínský and Blaťák [2005] also applied Inductive Logic Programming on the manual syntactic annotation and enriched the data by using WordNet. It is interesting to note that, even with this manual syntactic analysis and the fact that some systems carried tests only on the easiest part of the test set, most of these systems (the case of Kim et al. [2010] is discussed below) perform worse than our simple machine learning approaches.

Recently, in another comparison [Zargayouna, 2010], our results compared with another system. That system uses a SVM algorithm with different kernels and different levels of linguistic information. In comparison to our shallow linguistic information, two main models were proposed. In one of them called MIG-noLemma, the authors linearly followed the sentence with considering words as it is. In another model called MIG-noSyntax they used lemmas of the words (as given in the corpus) to train a SVM system. The results for those systems shown in the Table 8.8 which show that for similar data (using lemma of the word with SVM), our model achieved better recall while the precision is almost the same. In MIG-noSytax, they used lemma of word in the sentence to classify the relation between two proteins which is similar to our model. We used bag of lemmas between two proteins in the sentence which shows better recall.

In another experiments, they use deep linguistic information including the path in syntax tree given in the corpus and achieved better results as reported in [Zargayouna, 2010]. In fact, the relations in our experiments were coarse-grained in which we only had three labels (see Section 8.3.1). In order to evaluate the LM-KNN model with fine-grained relations, we utilized our model on the same data, LLL05 corpus, but with more detailed relations (10 different directed relations) including ActionTarget, BindTo, Interaction, PromoterDependence, PromoterOf, RegulonDependence, RegulonMember, SiteOf, TranscriptionBy and Transcrip-

Algorithm	**Precision**	**Recall**	**F-measure**
MIG-noLemma	0.52	0.2	0.29
MIG-noSyntax	0.55	0.29	0.38
libSVM linear kernel	58.0	56.6	57.3
LM-kNN	**82.2**	**80.3**	**81.2**

Table 8.8: Recent challenge results by training SVM with shallow linguistic information

tionFrom [Mondary and Zargayouna, 2011]. The evaluation results showed lower performance (precision=59.06%, recall=51.37%, f-measure=54.95%) with these data with fine-grained relations. According to the detailed error analysis results, the LM-KNN model could not classify some relations, which can show the limitation of our model to classify more detailed relations.

8.4.3 Discussion

With the development and the availability of powerful machine learning systems, many NLP problems are now modeled as classification tasks. As with our Random Forest or SVM experiments, such approaches usually yield good results. Yet, when taking into account the specifics of the task and the data, a huge improvement can be expected.

As the performance of our LM-kNN approach suggests it, lazy learning approaches (where the final decision is postpone to when the test data is ready) combined with simple tools like language modeling can offer an interesting alternative to complex tools, especially when dealing with small dataset and a complex classification task.

Another advantage of using a lazy-learning approach such as LM-kNN is that it may offer more robustness than model-based learning approaches when dealing with few examples. And if one wants to reduce the cost of the development of a relation extraction system, it is interesting to see how few examples are necessary to yield good enough results. Figure 8.4 shows the evolution of f-measure of our LM-kNN system on the LLL test set according to the number of interactions given as examples. For comparison, we also report the result of the rule-based system RelEx [Fundel et al., 2007], which was up to now the best performing system for this task (on raw data, but only tested on interaction without co-references). The evolution of the LM-kNN performance describes an expected curve: important variations are noticed when dealing with very few examples, the improvement is more important when adding examples to a small set of examples, and then the improvement is getting smaller; yet it is interesting to note that the curve

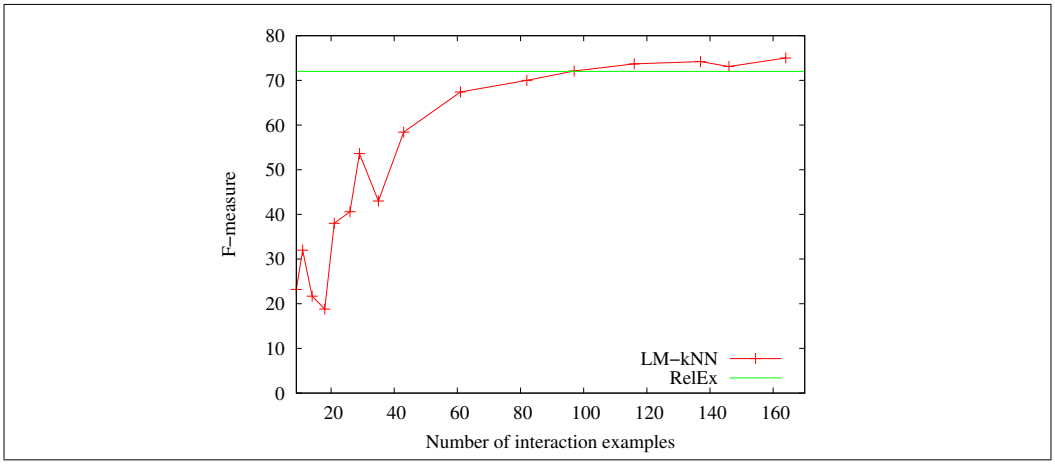

Figure 8.4: F-measure according to the number of interaction examples

suggests that more examples could still improve the f-measure of the system. The performance of RelEx is reached by our technique with less than 100 examples. Therefore, it suggests that instead of hand-crafting complex extraction rules that cannot be adapted to another extraction task, annotating only 100 examples is enough, which corresponds to about 50 sentences.

Indeed, recent studies rely on the syntactic path between the two entities on which either hand-written extraction rules are applied [Fundel et al., 2007, for example] or specially suited machine learning algorithms like string kernels or walk-weighted subsequence kernels for SVM are trained [Kim et al., 2010]. The results obtained are promising, yet, as we pointed it out before, they are highly dependent on the availability and the quality of the syntactic analysis (see [Fayruzov et al., 2008b]). For instance, the f-measure of Kim et al. [2010] declines by 15% when moving from a manual, perfect syntactic annotation to an automatic one. Moreover, long dependencies like the ones caused by co-references are seldom correctly detected. It is unfortunate that these syntax-based approaches where not evaluated on the more realistic test set including co-references. In this respect, having a co-reference resolution step could of course benefit to our language modeling approaches, but the small difference observed in the two experiments suggests that, here again, such a deep analysis is not necessary.

8.5 Conclusion

Extracting relations between entities is one of the most important parts of any Information Extraction systems as explained in Section 7.1.2. In this chapter we have presented and experimented several systems, that can be easily imple-

mented, to extract relations among entities in textual data in a supervised way. We evaluated our models on bio-medical text that has similar properties to transcribed text in terms of sentence complexity (because of having many unknown words), but also offers a good framework to evaluate our approach and compare its results with the state-of-the-art systems. Among all Information Extraction tasks in bio-medical field, we have chosen Protein-Protein Interaction extraction as a Relation Extraction task. We have shown that modeling this task as a classification problem and simply using shallow linguistic information is sufficient to reach good results. More precisely, we have represented a relation between two proteins or genes with a simple bag of lemmas that is later used by SVM or Random Forest classifiers, while the direction of the interaction is tackled as a 3-class machine learning problem. While not specifically new, this simple framework does not seem to have been tested in this domain.

Moreover, we have proposed a simple yet very efficient Relation Extraction system, LM-kNN, based on language modeling which better takes the specifics of the task and data into account. The results, evaluated on a publicly available dataset, underlined the interest of using shallow linguistic information and our new LM-kNN method yielded the best known results. The proposed model was also evaluated in a Quaero evaluation and out-performed other systems of PPI extraction.

From a technical point of view, it is possible to integrate these machine learning frameworks into an iterative process: newly retrieved relations are used as additional examples to re-train a system. Such approaches, like the one of Hearst [1992], as well as active learning techniques are of course straightforward for our lazy learning approach.

From an applicative point of view, our LM-kNN has to be tested over other Relation Extraction tasks. Indeed, our model, as developed, seems to be adequate for our ultimate goals in multimedia annotation and information extraction. In particular, we foresee its use for the detection of relations in speech transcripts of sporting events. As it was previously said, shallow linguistic approaches is a necessity in such a context in which the oral characteristics and the speech-to-text process prevent the use of any deep linguistic analysis tools.

So far, we assumed that relations are defined and annotated in data; but there are other cases in which this assumption is not valid or it is hard to define the type of relations. More simply, we do not have a prior knowledge about relations and want the system to automatically discover relation classes. In these cases, we need to discover relations instead of classifying them. This is the main reason why, in the next two chapters, we need to move from supervised models to unsupervised. Moreover, based on our goals in this thesis, we change the evaluation framework from bio-medical text to more related textual data. In Chapter 9, a relation discovery model is explained and discussed. Relations exist between entities in

a sentence. Detecting or extracting entities is another important part of any Information Extraction systems. To discover entities, we propose an unsupervised model for discovering entities form text in Chapter 10.

Chapter 9

Relation Discovery

9.1 Introduction

In the previous chapter, we proposed a supervised model for Relation Extraction by modeling it as a classification problem. It was based on the use of a shallow linguistic analysis of text to estimate the similarity between relations and an instance-based learning model to classify relations. We showed the effectiveness of Language Modeling in such a context. Supervised Relation Extraction models are useful for those problems that we have pre-assumption about the relations; when the type of relations are pre-defined. But in some cases, we are interested to find out relations among entities without any assumption about the type of relation. In this chapter, we focus on unsupervised approaches to discover relations among entities. According to our experiments in the previous chapter, we assume that shallow linguistic information can be effective in unsupervised learning too.

Different Machine Learning approaches are used to extract relations from textual data which can be grouped according to the level of supervision of the algorithm. Apart from the level of supervision, different levels of linguistic information are used in these models from shallow to deep linguistics information (for more details see Section 7.3.1).

The state-of-the-art approaches are mostly fully supervised which means that they need to have labeled data in order to train a model (see Chapter 8 for more details). These models have shown good results in closed domain data but still have problems with open domain data such as the Web. Besides the good results of supervised models, there are some challenges that make these models difficult to use for some applications. The difficulty of preparing the training data and biased trained models are two most important challenges in all supervised models. A supervised model can be biased because of algorithm problems or a non-appropriate training data.

On the contrary, fully unsupervised approaches do not need to have labeled

data. Considering relation discovery as an unsupervised model, there is no need to have prior knowledge about the data as it is assumed in this chapter. As far as it is related to the task of Relation Discovery, we need to know which features of objects are important to detect and cluster relations. Two different kinds of analysis can be used to define a relation in a sentence, deep/shallow linguistic analysis and probabilistic analysis (as explained in Section 7.3).

From a linguistic analysis point of view (as mentioned in the previous chapter), different levels of linguistic information are used in different Machine Learning approaches, from deep to shallow linguistic information. State-of-the-art systems use deep linguistic information [Zhang et al., 2005; Chen et al., 2011], but manually building deep linguistic information is expensive while automatically prepared linguistic analysis can decrease the final performance [Kim et al., 2010]. Moreover, for informal text, such as transcribed text, using deep linguistic analysis tools is not a good option when the results of the analysis is not reliable. For example, in a transcribed text of a football match, there are some non-word tokens which cannot be analyzed correctly within a deep linguistic analysis. For such texts, shallow linguistic information including words (or sequence of words) is more useful because it is less expensive and leads more reliable analysis.

In this chapter a new model for Relation Discovery, with minimum prior knowledge about the data, is introduced. More precisely, we model the relation discovery as a relation clustering in which the relations are defined between proper nouns. As it is previously explained (in Section 7.2.2.4), calculating the similarity among objects is an essential part of each clustering algorithm. In this chapter, we introduce a new similarity function, as the contribution of this chapter, which improves the clustering performance.

In the following sections, first, we review related work for unsupervised Relation Discovery. Then, we explain the proposed model in Section 9.3 and introduce the new similarity function based on an probabilities average of having a relation as a set of n-grams. To evaluate the model, we describe some experiments of this model on textual football reports (Section 9.4). Some analyses and a discussion about the experimental results are provided and followed by a conclusion.

9.2 Related Work

Relation Discovery is a task of mining text to discover relations without any assumption about which kind or number of relations we may find. Most of the Relation Discovery researches are based on discriminative models in which a feature set is used to calculate the similarity between objects (e.g. entities or relations). Then, these similarities are used with a clustering algorithm. The second (unsupervised) method is to use a generative approach to model the data,

then generalize it. In the following, we review some popular and state-of-the-art researches first on discriminative then on generative models.

As a classical way of discovering relations among named entities (NEs), Hasegawa et al. [2004] used Bag-of-Words between NEs as feature, weighted with a tf-idf score. Then, they clustered relations according to the cosine similarity between relations feature vectors. In this research, a potential relation is defined as the co-occurrence of two NEs in a sentence, if the number of words in between is less than a threshold. In order to reduce the noisy data, all pairs of NEs with low frequency are eliminated. After calculating the cosine similarity between each pair of relations, a hierarchical clustering algorithm with a complete-linkage strategy (more details in Section 7.2.2.1) is used to cluster relations. Finally, if most of the NEs in the same cluster have words in common, the common words are considered as the label for the cluster.

In addition to using Bag-of-Words (BoWs) between two entities, parse tree as a deep linguistic information is also used for relation discovery by Zhang et al. [2005]. Indeed, there are two main problems in BoW and cosine similarity in Hasegawa et al. [2004] model. First, they assumed that the same entity pairs in different sentences are linked by the same relation. Second, the flat feature vectors based on the words between two entities are insufficient to capture all the properties of a relation. Zhang et al. [2005] proposed to use the similarity between two parse trees in order to solve these problems. This similarity is defined as a function of the similarities between different properties of the parse trees. These properties are node tags (POS of the node for a leaf and linguistic category for non-leaf nodes), head words (exact word in the leaf and propagated word for non-leaf nodes) and entity types (PER (person) or COM (company) or GPE (geo-political entities)) and the relation order which determines in which direction the relation exists. Finally, a label is considered for each cluster based on the most frequent head word of the root node in the cluster. The proposed model shows good results both for frequent and less frequent entity pairs in the experimental results.

A relation can be expressed extensionally by considering all the instances of that relation or intentionally by defining all the paraphrases of that relation [Bollegala et al., 2010]. For example, consider the *acquisition* relation between two companies. An extensionally definition of acquisition includes all pairs of company names in which a company is acquired by another such as (YouTube, Google) or (Powerset, Microsoft). Intentional definition of the acquisition relation contains all lexical patterns that express that relation, such as *X is acquired by Y*, or *Y purchased X*, where X and Y are company names. Bollegala et al. [2010] proposed a model to co-cluster intensional and extensional representations of relations. In this model, only shallow linguistic information is used which is automatically generated by a Part-of-Speech tagger and a chunker. This informa-

Sentence	another example of a statutory merger is software maker Adobe Systems acquisition of Macromedia .
Substitution	Adobe Systems = X, Macromedia = Y
POS sequence	DT NN IN DT JJ NN VBZ NN NN X NN IN Y .
lexical patterns	X acquisition of Y, software maker X acquisition of Y, X of Y, software X acqusition Y
syntactic patterns	X NN IN Y, NN NN X NN IN Y, X IN Y, NN X NN Y

Table 9.1: POS sequence, lexical and lexical-syntactic patterns [Bollegala et al., 2010]

tion is used in three ways: POS sequence, lexical patterns and syntactic patterns which are listed in Table 9.1.

Considering all entity pairs as E and all extracted lexical-syntactic patterns as P, these authors proposed a sequential co-clustering algorithm to simultaneously cluster E and P. The first step is to build a matrix A in which each entity pair and lexical-syntactic patterns are represented as a row and a column respectively. Each cell of the matrix, A_{ij}, is the number of times the lexical-syntactic pattern p_j was extracted for the entity pair e_i. Thus each normalized row e_i in A is the distribution of an entity pair e_i over lexical-syntactic patterns. Likewise, each normalized column p_j is the distribution of a lexical-syntactic patterns over entity pairs. The co-clustering algorithm begins by sorting both rows and columns in A. Then, two empty clusters C_E and C_P are defined for entity pairs and lexical-syntactic patterns respectively. At each step of the iteration, first one pattern is extracted from the column of patterns to decide if it can make a new cluster or has to join the most similar column based on a threshold. The same model is defined to cluster entity pairs after each step of pattern clustering. Finally, when all patterns and entity pairs are clustered, the algorithm stops. Most co-clustering algorithms need to have the number of rows and columns in the matrix in advance which is not a requirement in the proposed algorithm.

In addition to discriminative unsupervised models for Relation Discovery, there are some researches that propose generative models. In these models, the goal is to learn from data and generate the results. Most of the generative models are based on a probability model.

A simple generative model for relation discovery can be a model based on the Latent Dirichlet Allocation (LDA) [Blei et al., 2003] in which named entities with the words in between can be considered as pseudo-documents in an LDA model. LDA, in general, considers a document as a mixture of topics that generate words with certain probabilities. This model can cluster (find topics for) pseudo-documents based on their words mixtures. But the problem is that the LDA-based

model considers all words equally. For example, distinguishing between general words (occurring frequently in the context of a relation) and trigger words (certain words in each relation) are not defined in this simple model. Moreover, there is no distinction between contextual words and the words that belong to the NE of the relation.

In order to solve these problems, Rink and Harabagiu [2011] proposed a model, called Relation Discovery Model (RDM), in which words are considered in two categories, relation trigger and general words. For example, in "*He developed* ***some air hunger*** *last night when I dropped* ***his tidal volume*** *from 450 to 350*", considering "*some air hunger*" and "*his tidal volume*" as a pair of interesting entities, *dropped* is a relation trigger word and other words in between are general words. Moreover, the entities are considered in a small set of semantic classes. For instance, *x-ray* is in medical test class and *lung cancer* belongs to a class of medical problem in *[x-ray] revealed [lung cancer]* sentence. They show that RDM can improve clustering results compared to a LDA model and traditional clustering methods.

Similarly, Yao et al. [2011] proposed a generative model based on LDA. More precisely, the proposed solution includes three generative models which are used sequentially to cluster the relations. The first LDA model uses dependency paths with two named entities to generate the first result for clustering. These simple features are not enough to capture all properties of relations so some clusters need refinements. The second LDA model is defined to take into account more features in order to split some clusters into more precise ones. For example, a cluster found by the first model contains *X was born in Y*, *X lives in Y* and also *X, a company in Y*. We know that the last sentence does not have the same relation as the others. So it has to be moved to a new cluster. In the second model, trigger words, lexical patterns, POS tag patterns and syntactic category pair features are added to the LDA. Assuming that relations can only hold between certain entity types, the third model, called type-LDA, is introduced to capture the selectional preferences of relations to their arguments.

There are some researches that focused on the importance of different aspects of the Relation Discovery. If we define the relation features as the contextual information (word between pairs of entities) for a relation and entity relation features as the contextual information for each instance of the entity, Rosenfeld and Feldman [2007] research shows that both relation and entity features are important to be considered in a Relation Identification (Discovery) task. Additionally, they considered a garbage cluster to collect all none-clustered relations since many of the relations do not belong to any cluster in final results. It is also reported that different filters can improve the unsupervised Relation Discovery performance [Wang et al., 2011]. Three heuristic filters, such as the number of words, the type and number of verbs between two entities, with a trained

statistical filter helped to filter most noisy and negative relations.

As it is explained before (see Section 7.3.1), a deep linguistic analysis is not robust to noise. Most of the models [Yao et al., 2011; Rink and Harabagiu, 2011; Bollegala et al., 2010; Zhang et al., 2005] presented above are based on deep linguistic analysis which cannot be an adequate solution for a robust Information Extraction system. Some models use a shallow linguistic analysis [Hasegawa et al., 2004] but still have some limitations because of using simple Bag-of-Words. Some state-of-the-art models, proposed for large scale relation extraction (or discovery) [Baroni et al., 2009; Etzioni et al., 2004], are not good for small data sets.

Previously (in Chapter 8), we showed the effectiveness of Language Modeling in Relation Extraction. In this chapter, we propose a model based on the idea of Hasegawa et al. [2004] but instead of simple Bag-Of-Words, we use n-grams in order to capture the sequential properties of the words in a sentence. Moreover, instead of using the classical cosine similarity function, we proposed a new similarity function based on Language Modeling and a probabilities average (see Section 9.3.3) to improve the results.

9.3 A Probabilistic Model for Relation Discovery

In our Relation Discovery problem, the goal is to semantically group relations among proper nouns in a sentence based on their contextual similarities. We model the problem as a clustering task where the goal is to assign a set of relations to the same group such that the relations within the group have more similarities to each other rather than to the relations in any other clusters. First, we need to define the similarity between relations based on their features. We consider contextual information as relation features so that each relation is represented as a vector based on contextual information (explained in Section 9.3.1). Then, different similarity functions are used to find the similarity among relations.

In our solution for discovering relations among proper nouns, first, we collect all potential relations from the text by defining each occurrence of proper noun pairs as a relation. It has already been shown that removing noisy data can improve unsupervised information extraction [Wang et al., 2011]. Similarly, we utilize some filters to remove non-relevant data and improve the data by eliminating pairs of entities without any relation for the potential relations. For example, the distance (i.e. number of words) between entity pairs is an important property for a relation. We assume that if there are more than a specific (threshold) number of words between the pair of proper nouns, we can remove the relation from the relation candidates. In order to study the importance of this constraint and find the best threshold, we examine different thresholds on data and analyze

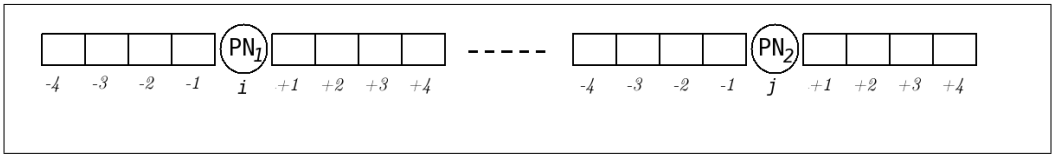

Figure 9.1: Entity pair and their contextual information

Relation	Contextual information (n-gram)
Fandel-Jurietti	[sanctionne] , [pour un tacle] , [un tacle trop]
Fandel-Totti	[sanctionne Jurietti pour] , [tacle trop appuyé] , [trop appuyé sur] , [.]
Jurietti-Totti	[Fandel sanctionne] , [pour un tacle] , [un tacle trop] , [tacle trop appuyé] , [trop appuyé sur] , [.]

Table 9.2: Two 3-grams around each pair of proper nouns

the results (reported in Section 9.4).

We consider relations and their similarities in a graph such that each node in the graph is a relation and the edge is the similarity between relations. First we build the similarity matrix A in which each cell a_{ij} is the similarity between relations r_i and r_j. Then, we use the Markov Clustering Algorithm to cluster the graph. The way of calculating the similarity matrix is the main contribution of this chapter.

9.3.1 Contextual Information

Different textual information sources are used for Relation Discovery, such as the path between entities in a parse tree [Zhang et al., 2005; Shinyama and Sekine, 2006] or the sequence of words or POS tags around entities in the sentence [Hasegawa et al., 2004; Wang et al., 2011] or even a combination of them [Bollegala et al., 2010]. We consider each occurrence of two proper nouns in a sentence as a potential relation. To define a relation, we use a contextual information as a sequence of words (n-gram) before and after the first and the second entity as depicted in Figure 9.1. For example, in the sentence "*Fandel sanctionne Jurietti pour un tacle trop appuyé sur Totti*", there are three proper nouns and three potential relations between them (without direction). The two 3-grams around the pairs of proper nouns for each relation are listed in Table 9.2.

We define contextual information in term of weighted n-grams. Since all n-grams do not have the same importance for the relation, we calculate its weight based on the tf-idf definition. We consider the sentences as the documents and the whole text as the corpus in the tf-idf definition (in Section 7.3.3).

In order to decrease data sparsity, all proper nouns are replaced by semantic class labels such as player names, referees, etc. Although, we use a manually built list of proper nouns with their class in this chapter, it is also possible to build the list automatically with the help of the techniques we propose in Chapter 10. Additionally, we use the lemmas of words instead of words, which decrease the data sparsity.

9.3.2 Filtering

In this work we are also interested to know which kind of information between entity pairs is important. For example, how the number of tokens between entities can help to improve the relation discovery results. Intuitively, when a relation exists between two entities in a sentence, we expect to have zero, one or two verbs (i.e. main verb + auxiliary verb) in between but not more. Similarly, we do not expect to have an entity name between pair of entities with a relation. It has been shown [Wang et al., 2011] that kind of filtering can improve performance. Wang et al. investigated different filters such as the number of words and verbs between entity pairs. In addition to those filters, we also consider another filter to eliminate entity pairs when more than a specific number of proper nouns exist in between. Moreover, we analyze the effect of each filter by itself during our experiments in Section 9.4. In Section 9.4.3, we analyze three filters in order to find out how important they are. However, using filters is a breach in the unsupervised approach, but it does not require any domain specific knowledge.

9.3.3 Similarity Functions

Previously in Section 7.2.2.4, we reviewed some classical similarity functions such as cosine and Jaccard. We have already shown (in Chapter 8) that a similarity function based on Language Modeling can perform better than classical similarity functions. In this chapter, we present a new similarity measure based on the probabilities average and Language Modeling.

Similarity based on the average probabilities: In addition to classical similarity functions, we propose a new similarity function which is based on the probabilities average of having one relation when the other relations are given. Considering R_1 and R_2 as two vectors (two sets of n-grams, representing two relations), we define the similarity between these two relations as the minimum of two average probabilities of having R_1 if R_2 is given and vice versa (see Equation 9.1). The minimum function helps up to avoid making long distance relations similar to many other relations. However, the problem of long distance relations can be solved by using distance filter as explained in Section 9.3.2

$$SIM(R_1, R_2) = \min\{F(R_1|R_2), F(R_2|R_1)\} \tag{9.1}$$

where $F(R_1|R_2)$ is a similarity measure based on the conditional probability of having R_1 when R_2 is given and calculated based on Equation 9.2. We use the minimum of the two average probabilities in Equation 9.1 in order to avoid high similarity for relations defined by longer sequence of words. In this equation, n-grams can be assumed to be random variables with a binomial distribution [Manning and Schütze, 1999, page 198] which makes the average probability reasonable for relation similarity function based on their n-grams. We define $NG_1 = \{ng_{11}, ng_{12}, \ldots, ng_{1r}\}$ as n-grams collected from relation R_1.

$$F(R_1|R_2) = \frac{1}{r}\sum_{i=1}^{r} P(ng_{1i}|R_2) \tag{9.2}$$

The conditional probability of ng_{1i} based on the given data R_2 can be easily calculated by a maximum likelihood estimation defined in Equation 9.3, where each n-gram is a sequence of words. For example with 3-grams, each n-gram in Equation 9.2 is defined as $ng = w_1w_2w_3$ and $c(w_1, w_2, w_3|R2)$ is the number of occurrences of sequence $w_1w_2w_3$ in R_2.

$$P(ng|R2) = \frac{c(w_1, w_2, w_3|R2)}{c(w_1, w_2|R2)} \tag{9.3}$$

In a probabilistic n-gram model, in order to prevent to have zero probability for unseen n-grams, we need to use a smoothing technique. Among all smoothing methods, we finally use the absolute discounting (see more details in Section 7.3.2.1), because it leads to good results in the Relation Extraction problem (Chapter 8). Moreover, it is simple to implement and does not require any training data. The smoothing technique used in this approach caused a minimum similarity between all pairs of relations which makes it difficult to have strong differences between relations. To solve this problem, we finally propose a discriminative similarity function. This function is inspired by Equation 9.1 but with a power to p (Equation 9.4).

$$SIM(R_1, R_2) = (\min\{F(R_1|R_2), F(R_2|R_1)\})^p \tag{9.4}$$

9.4 Experiments

In order to assess the proposed model, we applied it on minute-by-minute football reports (see more details in Section 9.4.1) to cluster relations among player names in each sentence. As for clustering algorithm, in the final set of clusters, we expect to have more similarity between relations within a cluster compared to

relations with other clusters. Among all clustering algorithms, we use the Markov Clustering Algorithm as a fast clustering algorithm which can cluster graphs. In the proposed model, we consider all relations as nodes in a graph which are connected together based on their similarities. By this definition, this graph can be considered as a simple weighted graph. Since we used smoothing in order to have the similarity between each pair of relations more than zero, this graph is a complete graph.

Finally, to evaluate the clustering results, the Adjusted Rand Index (ARI) is used (see Section 7.2.3 for more information). This evaluation metric needs to have a ground-truth which is available for our data as explained in Section 9.4.2. It is important to note that some relations only have relatively a small similarity with other relations; the clustering algorithm consider them as separate items which do not belong to any cluster. We decide to put all these not clustered relations in a new cluster for the final evaluation.

In the following sections, first we describe the data and review some statistics on it; then some experiments with the proposed model are compared with classical models to show how effective it is. Finally, some analyses of the results are provided and discussed.

9.4.1 Data

Minute-by-minute football reports collected from the Web (by the author) are used as data to evaluate the proposed model. An example of these reports are shown in Table 9.3 for two minutes. Then, Tree-tagger [Schmid, 1994] is applied to tokenize and find the Part-of-Speech tags for each word in the report. To build the ground-truth, the collected data were annotated by experts [Fort and Claveau, 2012] so that, for each report, we know the proper nouns with their semantic class (player, referee, coach, etc.) and the relations between them including *FairePasse* (passing the ball), *FaireFauteSurJoueur* (Doing foul on a player), *TaclerFaute* (foul tackle), *RemplacerJoueur* (player replacement), *Faire-Tentative2Passe* (multiple ball passing). A sample sentence with its POS tags and information from the ground-truth (i.e. proper noun semantics class and their relation) is listed in Table 9.4. All annotations were provided in an xml file separated from the text file.

We considered 4 minute-by-minute football reports of one team (bordeaux) which include 605 sentences. Considering, each entity pair as a candidate for relation clustering, we found 632 candidates. Among all these candidates, 134 are positive (exists in the ground-truth) and 498 are negative (absent in the ground-truth) relations (see the definition in Section 9.4.2).

Minute	Report
80	Zigic donne quelques frayeurs à Gallas et consorts en contrôlant un ballon chaud à gauche des 16 mètres au devant du Gunner. Le Valencian se trompe dans son contrôle et la France peut souffler.
82	Changement opéré par Raymond Domenech avec l'entrée d'Alou Diarra à la place de Sidney Govou, pour les dernières minutes. Une manière de colmater les brêches actuelles ?

Table 9.3: Minute-by-minute football report in French

Sentence	Perrotta cède sa place à Pizarro .
POS tagged	NAM VER:pres DET:POS NOM PRP NAM SENT
Proper Nouns	Perrotta→player and Pizarro→player
Relations	RemplacerJoueur(Perrotta , Pizarro)

Table 9.4: A sample sentence with POS tags (from TreeTagger) and a relation between two proper nouns

Relation	Frequency
No_Relation	198
R_FairePasse	23
R_FaireFauteSurJoueur	10
R_RemplacerJoueur	9
R_FaireTentative2Passe	4
R_TaclerFaute	2
Total relations	246

Table 9.5: General ground-truth

9.4.2 Ground-truth

A ground-truth with a list of relations with their class label, is needed to evaluate the clustering algorithm. We used two definitions of ground-truth in our experiments which are prepared based on the labeled data of minute-by-minute football report annotated by experts [Fort and Claveau, 2012].

In the first definition, we considered all pairs of entities in a sentence as a potential relation. If the relation exists in the ground-truth, it is a positive relation (whatever the real label is), otherwise it is considered as negative relation. With this definition, 246 relations with 6 different labels are found (see Table 9.5). Each of these ground-truth labels are explained in Table 9.6.

In addition to this general ground-truth which is useful to evaluate the proposed model with all data, in each experiment, we generate a second ground-truth as a subset of the general ground-truth. In different experiments, we use different subsets of data. The second definition of ground-truth contains only the relations that are used for clustering. This ground-truth helps to evaluate each experiment, only on the data that are used for modeling, in addition to the global evaluation based on the first definition. The differences between the role of these two evaluations are explained later in Section 9.4.4.

Clearly, the data is not balanced; it may bias the algorithm to the most frequent relations (in this case, negative relations). It is useful to filter out some negative relations from the data in order to improve the clustering. To do so, we define some filters to reduce the negative or noisy relations.

9.4.3 Filter Analysis

The following four filters are used to cut down the noisy or negative relations. Each filter has its own parameters which need to be estimated. In order to do that, we use the ground-truth to find out the best parameter estimation. To

Relation name	**Description**
No_Relation	All negative relations are defined by this label
R_FairePasse	One player passed the ball to another player
R_FaireFauteSurJoueur	One player did a foul on another player
R_RemplacerJoueur	One player is replaced by another player
R_FaireTentative2Passe	In the case of having multiple ball passing
R_TaclerFaute	A foul tackle from on player on another

Table 9.6: Relations definition in the ground-truth

verify the parameter estimation, the results of this estimation is compared with the results with the best model.

Player-player filter is a filter defined specifically for the minute-by-minute football reports. Analyzing the ground-truth shows that relations only exist between players and not between other pairs of proper nouns. This observation can help to reduce the negative examples from the data. We only need to cluster a relation if it is between pairs of players. Other relations can be considered as negative relations. So from this point, whenever the relation is used, it means a relation between two players.

Distance filter is defined based on the number of tokens existing between the two proper nouns in a sentence. The ground-truth analysis shows that the positive relations do not exist between pairs of players when the distance is more than sixteen tokens. In Figure 9.2, the frequency of positive and negative relations with different distance thresholds is shown. For example, for pairs of players with less than 13 tokens in between, there are 414 negative and 132 positive relations.

It is also useful to know with which distance threshold the data is more balanced in term of positive and negative relations. The ratio of positive and negative relations versus distance can help us to find the best threshold to have a better balance of the data. Relation coverage is defined in Equation 9.5, as the subtraction of the positive relation ratio from the negative relation one which is depicted in Figure 9.3 in orange.

$$\text{relation coverage } = \left| \frac{\text{positive relations}}{\text{total relations}} - \frac{\text{negative relations}}{\text{total relations}} \right| \tag{9.5}$$

The relation coverage shows that the best distance to have relatively more positive relations and less negative relations is for $distance <= 6$. Thus, we expect to have good results near to this distance which can cover about 38.15% of all positive relations and 74.63% of all negative relations. The data is still not balanced with this threshold (34.48% vs. 65.52% for positive and negative relations respectively) but is more balanced compared to the data without filtering (21.20% vs. 78.80%).

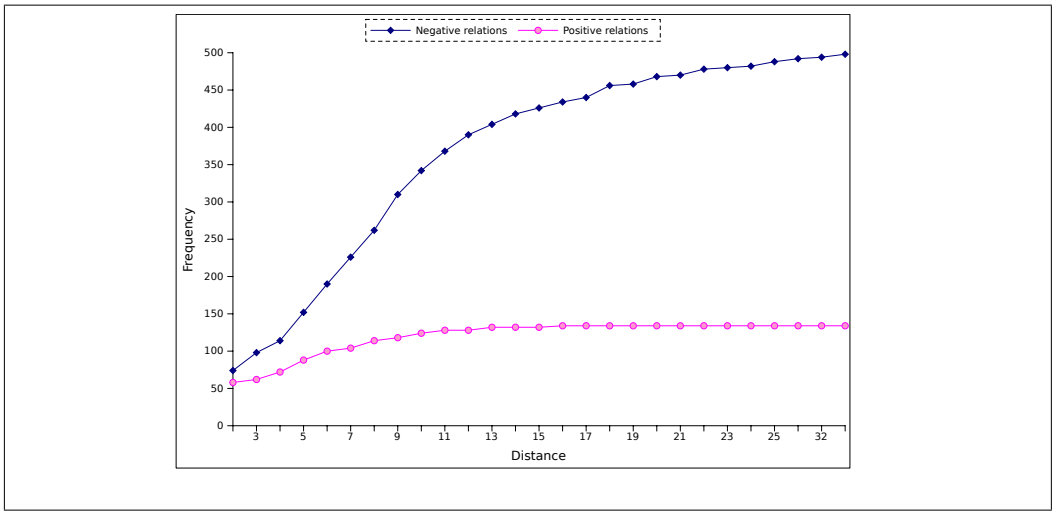

Figure 9.2: Distance vs. positive and negative relations

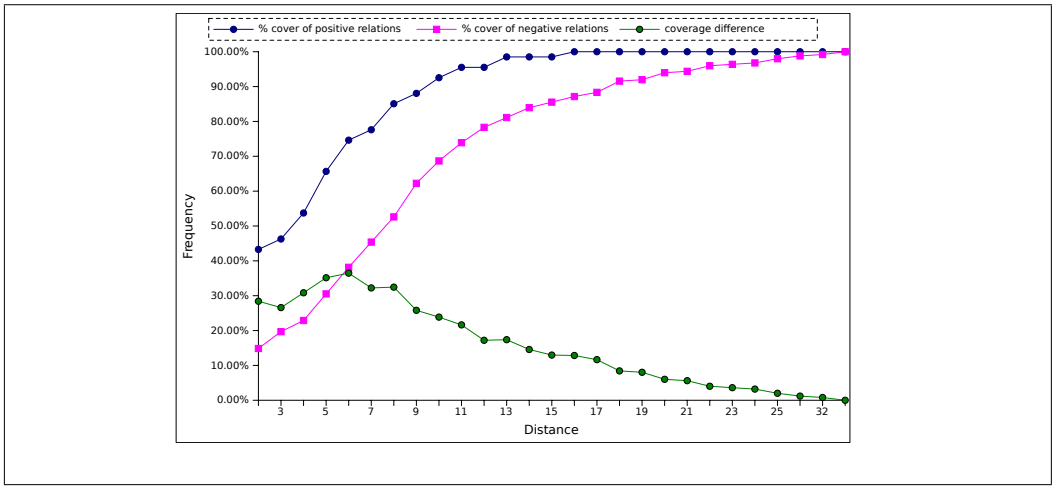

Figure 9.3: Positive and negative relations ratio vs. distance

# of PN	Relation frequency	Coverage
0	133	69%
1	43	22%
2	12	6%
3	3	2%
4	1	1%

Table 9.7: Number of player names between pairs of entities for negative relations

Frequency	Positive	Negative	Positive coverage	Negative coverage
0	26	61	40.00%	31.77%
1	28	55	83.08%	60.42%
2	8	32	95.38%	77.08%
3	3	29	100.00%	92.19%
4	0	13	100.00%	98.96%
5	0	2	100.00%	100.00%

Table 9.8: Positive and negative relations frequencies vs. number of verbs between entity pairs

Player name filter is another useful filter based on the number of player names between two other players which helps us to remove noisy or negative relations from the data. The analysis of ground-truth shows that most of the positive relations do not have any other player name in between. For positive relations, among all 65 relations, only one of them has another player name in between. But for negative relations, the distribution of player names between entities is different as shown in Table 9.7. Based on this analysis, we decided to keep all relations that do not have any other player name between entity pairs in the final data. This filter helps to have 30% less negative relations and more balanced data.

Verb filter depends on the number of verbs between pairs of players which has already shown improvement in similar task [Wang et al., 2011]. Similarly to previous filters, we analyze the ground-truth with this filter to have a better estimation of the number of verbs between pairs of entities to extract potential relations (see Table 9.8). Clearly, when the number of verbs between entity pairs are more than 3, there is no positive relations in the data. This filter can help to reduce negative relations while positive relations are unchanged (or slightly changed).

All these filters can affect the balancing of the data in terms of negative and

positive examples. For example, with verb filter, if we filter out all relations with more than 3 verbs between pairs, it means that all positive relations are kept and 7.81% of negative relations are filtered out (see Table 9.8). In the following section, we report different experiments to show how these filters may improve the Relation Discovery results.

9.4.4 Results

Different experiments are designed to evaluate different aspects of the proposed model including the filtering. In each experiment, we fix some parameters, then change the others to see their importance for the problem. In all cases, Filter1 is about the limitation on the number of player names in between, Filter2 is defined as the limitation on the number of tokens in between and Filter3 is considered as the number of verbs between entity pairs. In all experiences, the clustering is done in two phases, filtering and clustering. In the filtering phase, we first filter the data (with one or more filters); then the Markov clustering algorithm is applied on the filtered data. In order to have evaluation results on the complete data, all filtered relations are considered as negative relations in the final evaluation.

9.4.4.1 Best Filter Combination

In this experiment, different combinations of filters are evaluated in addition to the evaluation of each filter separately, and the goal is to find the most effective combination of filters. Two different evaluations are reported in Table 9.9. The first ARI is the evaluation of the algorithm when the filtered relations are not in the final clustering results. Since the number of relations in the final results are not the same, the comparison cannot be correct. We use the idea of Rosenfeld and Feldman about the garbage cluster, but instead of ignoring them, we collect them as a new cluster. The second ARI (ARI_2) is the evaluation results when all filtered relations are considered as a new cluster (merged with garbage cluster) in the final results. So in all experiences, the number of relations are the same in the final results which makes the comparison reasonable. It was expected to have a higher ARI_2 when the filtered relations are considered as a new cluster because most of those relations belong to negative relations (based on the ground-truth).

Finally, we considered one 3-gram after the first named entity and one 3-grams before the second named entity as the contextual information. In all of the following experiments, the similarity function is defined based on the probabilities average as explained in Section 9.3.

As explained in Section 9.3.2, the goal of the relation filtering is to reduce the noisy or negative relations. In other words, an effective filter is the one that can improve the clustering with the minimum elimination of positive relations and

W3-N3	**ARI**	**Relations**	**Clusters**	ARI_2
No filter	1.46	246	40	1.46
Filter1	12.43	190	19	18.88
Filter2	8.39	179	29	14.35
Filter3	4.83	205	32	8.64
Filter1 + Filter2	15.47	154	17	21.61
Filter1 + Filter3	**19.40**	165	15	26.55
Filter2 + Filter3	8.25	173	25	15.23
All filters	**19.77**	148	15	**27.23**

Table 9.9: Evaluation of different combinations of filters with the probabilities average similarity function

maximum cutting off the negative relations.

In this experiment, the baseline model is defined as the clustering of the data without filtering, called *no filter*. Among all filters, based on the results in Table 9.9, the most effective filter is Filter1 (no player name in contextual information) that improves the *no filter* model clearly more that two other filters.

Assuming noisy relations as the relations without strong common patterns with other negative relations, improving the results with Filter1 shows that the importance of the number of player names in between and helps to remove noisy relations from the data. Most of the positive relations are important because of their common patterns between relations in the same cluster; but for negative relations, there are various patterns which can cause the bad performance. In addition to this argument, it was also expected to have this improvement with the Filter1 because the eliminated (negative) relations are considered in the final evaluation as a new cluster.

This experiment also shows that filtering relations based on the number of verbs in between is useful, which is reasonable. It is because, mostly, a relation is defined by one verb or a combination of two verbs (auxiliary verb + main verb). In Filter2, all relations with more than 2 verbs in between are removed before using the data for clustering.

The distance between two player names is considered fixed to 10 in order to study the role of this filter. This experiment did not show acceptable improvement based on the number of tokens between two player names by itself (Filter3); but the combination of this filter with other filters shows improvement. For example, the best ARI with Filter1 is 18.88 which is improved to 26.55 when combined with Filter3. Combination of all filters shows the best results which is reasonable since most of the negative relations are eliminated in the filtering phase and build a new cluster at the end.

9.4.4.2 Contextual Information Experiment

We are also interested in finding out which parts of the contextual features (as explained in Section 9.3.1) are more important. Contextual information is considered in 4 groups according to the position relative to the first or the second player name in the sentence. It includes sequences of words (n-grams) before and after the first and the second player names. In this experiment, we evaluate different combinations of contextual information with the best model (all filters) that we found in the first experiment. The results show that using all the contextual leads to very poor results but the contextual patterns in between (R_{12}, R_{13}, R_{22}, R_{23}) give good results. The most important patterns are between player names and outer contextual information (R_{11}, R_{14}, R_{21}, R_{24}) can increase the similarity between many pairs of players.

In order to calculate the similarity between two relations, the sum of the similarities between the corresponding parts of contextual information is used as defined in Equation 9.6.

$$sim(R_1, R_2) = \sum_{i=1}^{4} SIM(R_{1i}, R_{2i}) \tag{9.6}$$

In this equation, R_{11} is the contextual information before the first player name in the first relation and similarly, R_{21} is the contextual information before the first player name in the second relation. Among different ways of computing the similarity between two relations, two ways (Figure 9.4) are examined in this experiment. The second way of computing the contextual similarity obtained very poor results. In the first way, the best results are obtained when only inner contextual information ($R_{12}vsR_{22}$ and $R_{13}vsR_{23}$) is used. In addition to the sum of different contextual information, we used the maximum and minimum of these contextual similarities. The maximum function (Equation 9.7) shows the *ARI* near to the best result when it considered only sequence of words between two entities. When we consider all four parts of contextual information the results were not better than a random clustering (ARI=0).

$$sim(R_1, R_2) = \max_{i=1}^{4} SIM(R_{1i}, R_{2i}) \tag{9.7}$$

9.4.4.3 Similarity Functions Experiment

This sub-section aims at comparing the proposed similarity function with classical similarity functions such as Cosine and Power Scalar [Ebadat et al., 2012] (see Section 9.3.3). In this experiment, a combination of the number of player names in between (Filter1 with no player in between) and verb filter (Filter3 with less than 3 verbs) is used. These two filters reduce the total of relations to 165.

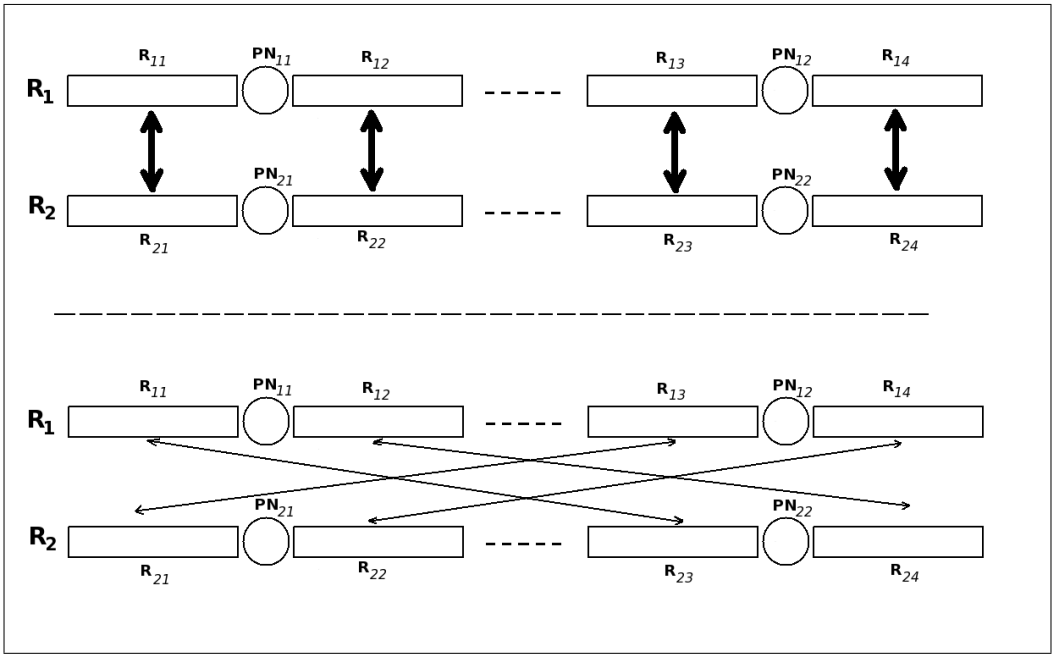

Figure 9.4: Two ways to calculate the contextual similarity between two relations

Similarity function	ARI	Relations	Clusters
probabilities average	19.4	165	15
Cosine + tf-idf	10.43	165	8
Power Scalar + tf-idf	9.12	165	11
Jaccard	13.44	129	8

Table 9.10: Comparison of different similarity functions

For all of the listed models, 3-gram of 3 words between two player names are considered as contextual features since it shows the best results compared to other combinations of n-grams and words. For cosine and Power Scalar, all n-grams are weighted based on the tf-idf model. Jaccard and probabilities average functions cannot use such weighted features.

The proposed similarity function (probabilities average) shows the best results compared to other classical functions. Clustering with Jaccard similarity causes some single member clusters. This is the reason why there are only 129 clustered relations in the final results with Jaccard. The remainings are ignored in final evaluation.

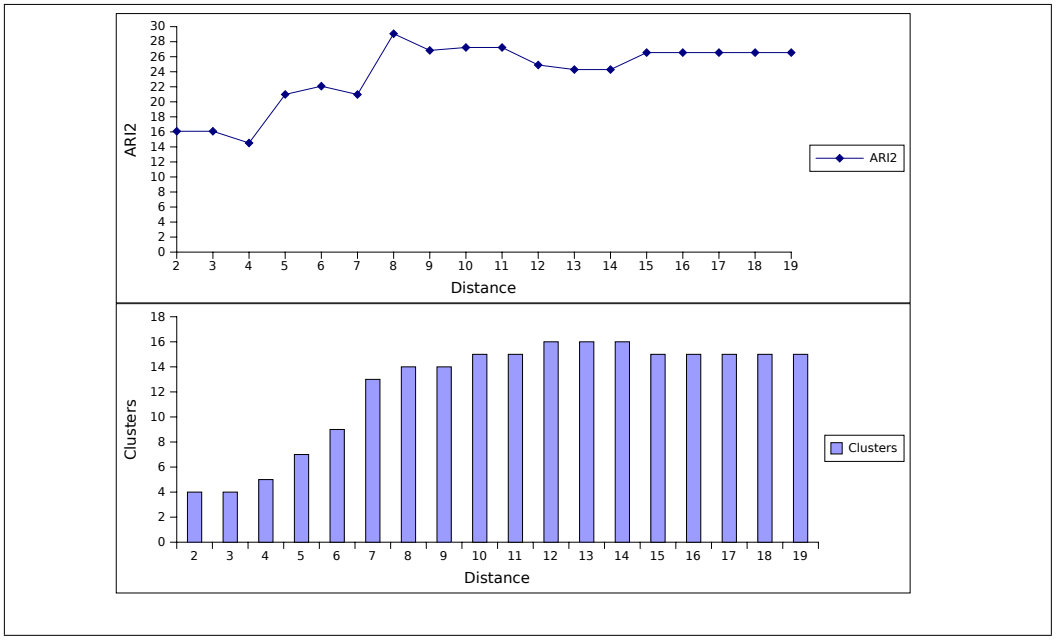

Figure 9.5: Distance analysis with ARI_2

9.4.4.4 Distance Filter Experiment

In this experiment, the goal is to find if the assumption about filtering relations based on the number of words between entity pairs is correct or not. Based on the graph in Figure 9.3, it seems that the best results could be obtained for a distance equal to six. In this experiment, we examine again this assumption by changing the distance in the best model including all filters defined in Section 9.4.3.

Filtering relations based on the number of tokens between two player names did not show improvement as shown in Figure 9.5. In addition, the ARI_2 and number of clusters are stabilized by increasing the distance. This result shows the importance of considering all relations regardless of their distance.

The results in Figure 9.5 are obtained by applying all filters and changing the distance in Filter2. It has already been shown that the distance filter is not effective by itself when compared with the model with all filters. Another experiment shows that in a model with only the distance filter, the results are similar to what we expect based on the filter analysis results (in Section 9.4.3). In this experiment, only the distance filter, with different value for d, is used with the proposed model. The results that are depicted in Figure 9.6 show similar trends than in Figure 9.3. In both figures, the best results are obtained with a distance of 6 and 8; increasing the distance cuts down the system performance. As a conclusion, the distance filter is effective by itself but there are other filters

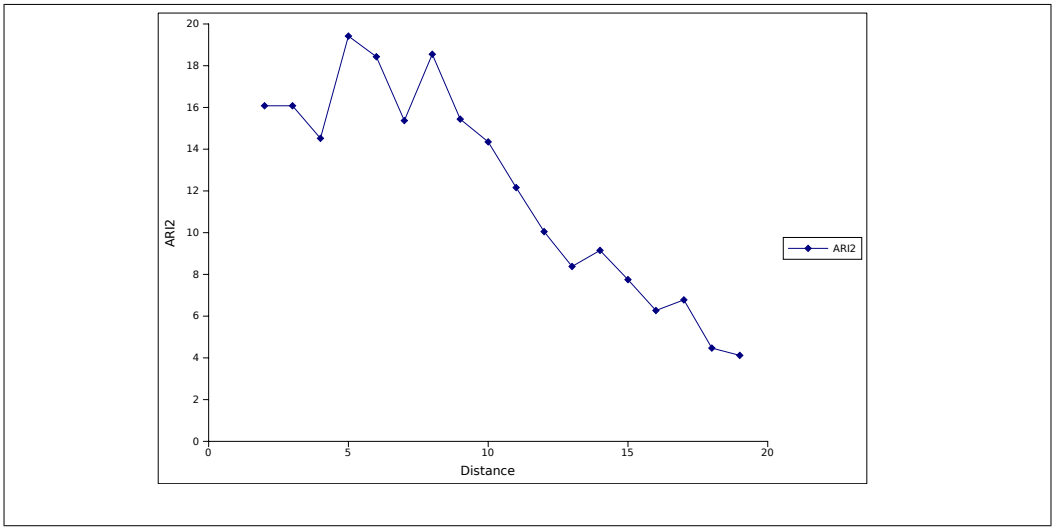

Figure 9.6: Analysis of distance filter - no other filters are used

that are more effective than it. In other words, verb filter and player-player filter cover distance filter cases by removing noisy or negative relations from the data.

9.4.5 Error Analysis

The results in previous sections show the importance of using the proposed similarity function based on the Language Modeling and probabilities average. But still, these results are not high. In order to investigate the causes of errors, we used the concept of B-Cubed [Bagga and Baldwin, 1998] evaluation metric. B-Cubed is a metric to calculate precision, recall and f1-measure for each cluster based on the ground-truth. The calculated metric for each cluster member can be useful also to estimate the performance for each class of members in the ground-truth. So, we used the average precision, recall and f1-measure for each relation in order to find the most problematic class of relations. Additionally, as we explained in Section 9.4.4, three evaluation levels have been defined for the final clusters.

In Table 9.11, all the evaluation results are reported for these three levels. In this table, *Elementary* means the first cluster evaluation (based on ARI) which does not necessarily cover all relations. Then, all single member clusters are considered as a new cluster (similar to the garbage cluster in Rosenfeld and Feldman) and the next evaluation, ARI is generated. Finally, the ARI_2 is estimated with all filtered relations merged in the garbage cluster. All of these values are f1-measures calculated based on the Equation 7.11 with $\beta = 1$.

Among all relations, only *TaclerFaute* shows no changes at different levels of the evaluation. Since there are only two instances in this class (see Table 9.5 for

Relation	**Elementary**	**ARI**	ARI_2
TaclerFaute	39%	39%	39%
RemplacerJoueur	74%	54%	54%
FairePasse	33%	33%	27%
FaireFauteSurJoueur	83%	46%	43%
No relation	26%	68%	81%
FaireTentative2Passe	57%	36%	29%

Table 9.11: F1-measures for different relations, for different clustering results

more details), this result is expected. The f1-measure decreased for the *RemplacerJoueur* relation from the elementary evaluation to ARI which means that the similarity function could not 100% separate this relation from the others. The reduction from ARI to ARI_2 shows that the filtering phase also filtered some relations from this class. The same arguments are correct for all other relations which show a reduction in each level of evaluation. Exceptionally, for negative relations (marked as *no relation*), the evaluation shows improvement at both levels. It means that the filtering and the similarity function can successfully detect negative relations.

9.5 Conclusion

In this chapter, an unsupervised model was proposed to detect and cluster relations in a sentence between entities. Here, we moved from a supervised model, presented in Chapter 8, to an unsupervised model. The proposed model was evaluated with data related to video and thus in a framework closer to the ultimate objective in which this thesis was proposed. Similar to other clustering algorithms, different similarity functions were examined in this research, including a proposed function based on the probabilities average and Language modeling. This new similarity function is the main contribution of this chapter because it improves the Relation Discovery performance compared to classical similarity functions. Briefly, experimental results show that the proposed similarity function based on probabilities average works better than classical similarity functions such as cosine and Jaccard. However, the final error analysis leads us to some drawbacks in the system. The model can separate negative and positive relations but still needs more efforts to improve the clustering of positive relations. There are some relatively large clusters that the model fails to split into smaller one. We assumed that our data is noisy, so we defined some filters in order to eliminated noises. Moreover, the result analysis showed the importance of the filters

to balance the data in terms of the negative and the positive relations.

One way to improve the system would be to define a second level of clustering for the larger clusters. Additionally, the data that is used in the experiments were small so there were some relations with low frequencies. Since in the proposed model the similarity is based on the common patterns between relations, experimenting the model on more balance data should help to yield better results. Using a combination of similarity measures instead of a single measure has been shown interesting results in Relation Extraction [Panchenko and Morozova, 2012] and may be useful for Relation Discovery too.

Chapter 10

Proper Noun Clustering

10.1 Introduction

In the two previous chapters, we proposed some models for Relation Extraction and Discovery a main module of Information Extraction system. In the last two researches, we assumed that entities are nominated in text; but detecting entities and classifying (or clustering) them is another challenging task in a robust Information Extraction system as it is aimed in this thesis. In this chapter, we are concerned with the clustering of entities extracted from texts, namely proper nouns, based on their contexts in a corpus. Note that this task is close to Named Entity Recognition (NER), but differs from it in some respects. Indeed, the goal in Named Entity Recognition is to locate and classify Named Entities into predefined groups such as Person, Location and Organization names. Locating and classifying could be done either in one step or in two consecutive steps, but for these two sub-tasks, most NER systems rely on supervised models, trained on manually tagged data. Yet, in this work, our goal is slightly different from this strict definition since we aim at building classes of entities without any supervision or presupposition about the classes. More precisely, we want to group proper nouns (PNs) into different clusters based on their similarities without a prioiri knowledge on the possible classes.

The choice of the similarity function is highly dependent on the representation used to describe the entities. In this chapter, we investigate the use of a new representation which is expected to outperform the one commonly used. Indeed, the classical way of calculating similarities is to build a feature vector, or Bag-of-Words, for each entity and then use classical similarity functions like cosine. In practice, the features are contextual ones, such as words or n-grams around the different occurrences of each entity. Here, we propose to use an alternative representation for entities, called Bag-of-Vectors, or Bag-of-Bags-of-Words following ideas partially proposed by Gosselin et al. [2007] in an image retrieval framework.

In this new model, each entity is not defined as a unique vector but as a set of vectors, in which each vector is built based on the contextual features (surrounding words or n-grams) of one occurrence of the entity in the corpus. The usual similarity or distance functions including cosine, Jaccard and Euclidean distances can be easily extended to handle this new representation. In this chapter, these various representation schemes and distances are evaluated on our proper noun clustering task.

In the next section, we review related work and then present the different representation schemes for our task, including the Bag-of-Vectors, in Section 10.3. The use of this representation to compute similarities and finally cluster the entities is presented in Section 10.4. Experiments are then reported in Section 10.5 for different similarity functions and feature vector models. Finally, conclusive remarks and foreseen work are given in the last section.

10.2 Related Work

Extracting and categorizing entities from texts has been widely studied in the framework of Named Entity Recognition. The history of NER goes back to twenty years ago; at that time, its goal was to "extract and recognize [company] names" [Nadeau and Satoshi, 2007]. NER is now commonly seen as the task of labeling (classifying) proper nouns or expressions into broad subgroups, such as person, location, organization names, etc. [Kim Sang et al., 2003], or more recently into fine grain groups (e.g., a location can be a city, a state or a country) [Fleischman and Hovy, 2002; Ekbal et al., 2010].

The differences between a NER system and our task is that we need to discover and cluster proper nouns in the corpus (not in the sentence). So the final result in a NER system is an annotated text and the final results is a clustered list of detected proper nouns based on their semantic roles in the corpus.

Several approaches are used for NER which could be organized in three main groups. The most common approach is the supervised one; it needs annotated data to train a supervised machine learning algorithm such as Support Vector Machine [Isozaki and Kazawa, 2002; Takeuchi and Collier, 2002], Conditional Random Fields [McCallum and Li, 2003a; Sobhana et al., 2010; McCallum and Li, 2003b], Maximum Entropy [Chieu and Ng, 2002] and Hidden Markov Model [Zhou and Su, 2002]. In these NER models, the quality of the final results chiefly depends on the size of the training data. Despite much effort to develop good supervised methods, most of them have scalability problems as the data grow or the labels become more precise such as professional for person names [Whitelaw et al., 2008; Ekbal et al., 2010].

A second approach is to use semi-supervised machine learning; it has received

Feature	**Description**
full-string=x	For example, with "*Maury Cooper*", full-string=Maury_Cooper
contains(x)	For multi-parts names, all parts have to be included. e.g., Maury Copper, *contains(Maury)* and *contains(Cooper)*
allcaps	It is true if all characters are capital letters. e.g., IBM
allcaps2	It is true if all characters are capital letters and contain at least one period (e.g., N.Y.)
nonalpha=x	Contains all non alpha characters in the entity (e.g., for A.T&T. nonalpha=&.)
content=x	The context for the entity. e.g., for *..., says Maury Cooper, a vice president at S.&P.*, context=*president*
context-type=x	The context type is *appos* for appositive case and *prep* in the PP case

Table 10.1: Features and spelling rules [Collins and Singer, 1999]

a lot of attention recently, especially when the annotated dataset is small or absent. Different models have been studied under this category including rule-based systems [Liao and Veeramachaneni, 2009] in which simple rules help to build some annotated data, then a CRF classifier, trained on these data, generates new training data for the next learning iteration. Kozareva [2006] uses some clue words in order to build a gazetteer lists from unlabeled data; this list is then used to train different NER systems.

Whether supervised or semi-supervised, these approaches rely on predefined group of entities (and the corresponding training data). Yet, in a context of information discovery, defining the interesting NE categories requires deep knowledge of the domain and may bias the systems since they focus on these categories and may miss interesting information. The last approach is the unsupervised one. To the best of our knowledge, there is no pure unsupervised NER system. Indeed, some systems claim to be unsupervised but either rely on few hand-coded rules [Collins and Singer, 1999], or external resources such as Wikipedia [Kazama and Torisawa, 2007].

In a rule-based (hand-coded) model, Collins and Singer [1999] applied seven predefined spelling rules (seeds) on unlabeled data (see Table 10.1); then, they extracted some contextual rules which are defined based on the maximization of having a context and a label. The contextual rules are ranked according to their strength, calculated by Equation 10.1 as the estimation of the conditional

probability of $p(y|x)$.

$$h(x,y) = \frac{count(x,y) + \alpha}{count(x) + k\alpha} \tag{10.1}$$

where $count(x,y)$ is the number of times that x, as a rule, has been seen with y, as a label in the labeled data. α is the smoothing parameter and k the number of labels (for example, 3 for *person*, *organization* and *location*). In supervised model of this estimation proposed by Yarowsky [1995], a more sophisticated smoothing technique is proposed; but it is not applicable for unsupervised model since it needs a training data to estimate α in equation. The new labeled data can generate a list of new spelling rules. In the next step, top ranked spelling rules have to be added to the rule list. These steps repeat until having the specific number of rules applied.

An unsupervised model proposed by Etzioni et al. [2005] can extract Named Entities from the Web but without solving the ambiguity among them. The final results depend on the output of Web search engines. But, having a list of entity/label is not enough to find the named entity labels in a given document. Nadeau et al. [2006] proposed a simple alias resolution algorithm in order to resolve the ambiguities in the following three cases which cannot be covered by a simple list of named entities:

1. **Entity-Noun** ambiguity which is mostly between plural words, such as "jobs" and the surname "Jobs".

2. **Entity Boundary** ambiguity mostly occurs between multi-word entities such as "Boston" and "Boston White Sox"

3. **Entity-Entity** ambiguity is between two different classes of the same word. For example between "Sydney" as a person name and city name.

In the group of unsupervised models, different approaches are proposed that rely on Wikipedia as a knowledge base. Kazama and Torisawa [2007] use Wikipedia to improve a Named Entity recognition system. The proposed model considers the first noun phrase after the first occurrence of *be* in the first sentence of a Wikipedia entry of the given named entity as a feature in a CFR-based NE tagger. Toral and Munoz [2006], instead, use all the noun phrases in the first sentence in Wikipedia for the given named entity to map them to a WordNet entry and find the abstract category. But, Wikipedia is not helpful for more detailed NE recognition systems where the goal is to find sub-types of people [Whitelaw et al., 2008].

All above models work well with formal text but switching from grammatical (formal) text to non-formal texts such as emails, blogs, and tweets can drop their performance. For example the performance (f1-measure) of the Stanford NE recognition system [Finkel et al., 2005], which was trained on the CoNLL03

shared task data set, drops from 90.8% to 45.8% [Ratinov and Roth, 2009] in such a case. Moreover, automatically generated POS tags may not be reliable for this informal text. For example, the state-of-the-art POS tagger accuracy drops to 74.0% on tweets [Liu et al., 2011]. So, NE recognition systems that are based on formal text linguistic analysis cannot survive with these new but huge amount of text data.

In this chapter, we proposed a new representation scheme which benefits from the instance of each entity in the corpus. From a technical point of view, similarity on complex objects (graphs, trees...) have been widely explored [Bunke, 2000]. Such representations and similarities are seldom used in information retrieval due to their computation costs. The Bag-of-Vectors representation that we propose to investigate in this chapter is inspired from the Bag-of-Bags are used for image classification with SVM by Kondor and Jebara [2003]; Gosselin et al. [2007].

Gosselin et al. [2007] proposed a general kernel framework for object retrieval embedded in images. They represent each image as a set of vectors (bags of features) and then define the kernel framework to classify objects in images. The proposed framework is a discriminative kernel which can increase the high matches between bags of different images. The Bag-of-Bags data representation was used for supervised Machine Learning such as classification. This representation is expected to be well suited for NLP tasks like ours while conserving manageable computational costs. Mostly, Bag-of-Bags data representation was used in a supervised Machine Learning context but, here, we use this model for an unsupervised Machine Learning algorithm to cluster entities in a text.

10.3 Representing entities with Bag-of-Words and Bag-of-Vectors

In this discovery of semantic classes, we focus on proper nouns (PNs) contained in French football reports. The texts are Part-of-Speech tagged using TreeTagger [Schmid, 1994], and the PNs are simply collected based on their tags. In order to cluster them, we need to represent these PNs so that similarities can be computed between them. As it was previously explained, a vectorial representation is commonly used for this type of task: a PN is represented by one contextual vector. In this chapter, we investigate the use of a new representation scheme, the Bag-of-Vectors, in which a PN is represented by several contextual vectors. In the remaining of this section, we first explain which contextual features, common to these two representation, are used. Then, we successively present the Bag-of-Words and Bag-of-Vectors approaches.

Sentence	
Zigic donne quelques frayeurs à Gallas et consorts en contrôlant un ballon chaud à gauche des 16 mètres au devant du Gunner.	
PN	**n-gram feature**
Zigic	donne quelques frayeurs — quelques frayeurs à
Gallas	donne quelques frayeurs — quelques frayeurs à, et consorts en — consorts en contrôlant
Gunner	mètres au devant — au devant du

Table 10.2: N-gram features for proper noun N=3, W=4

10.3.1 Contextual Features

Different contextual features were explored in our experiments, based on words, lemmas or n-grams surrounding each occurrence of a PN. In the experiments reported in this chapter, we only present the results for the features that yielded the best results. They are based on 3-grams collected in a window of 4 tokens before and after each PN occurrence in a sentence. An example of collected n-grams is given in Table 10.2.

Different weighting schemes for the collected n-grams were also explored (section 7.2.2.4), in order to give less importance to very common ones. Here again, for simplicity purpose, we only present the one giving the best results, which is a modified tf-idf [Momtazi et al., 2010] in which the idf score is calculated for sentences instead of documents (note that in a short window, tf is almost always equal to 1; the weighting scheme is thus mostly a pure idf).

10.3.2 Bag-of-Words (BoWs)

In the standard BoW model, for each PN detected in the corpus, a single (weighted) feature vector is built based on the n-grams before and after all the PN occurrences in the whole corpus. Thanks to its sparsity, the resulting vector allows very effective distance computation. Yet, in such a representation, the n-grams coming from the different occurrences of a PN are mixed (see Figure 10.1). Thus, based on this representation, the comparison of two PNs cannot be made at the occurrence level. The Bag-of-Vectors representation that we propose to use, aims at keeping the good properties of the vectorial representation, while offering an occurrence-based representation.

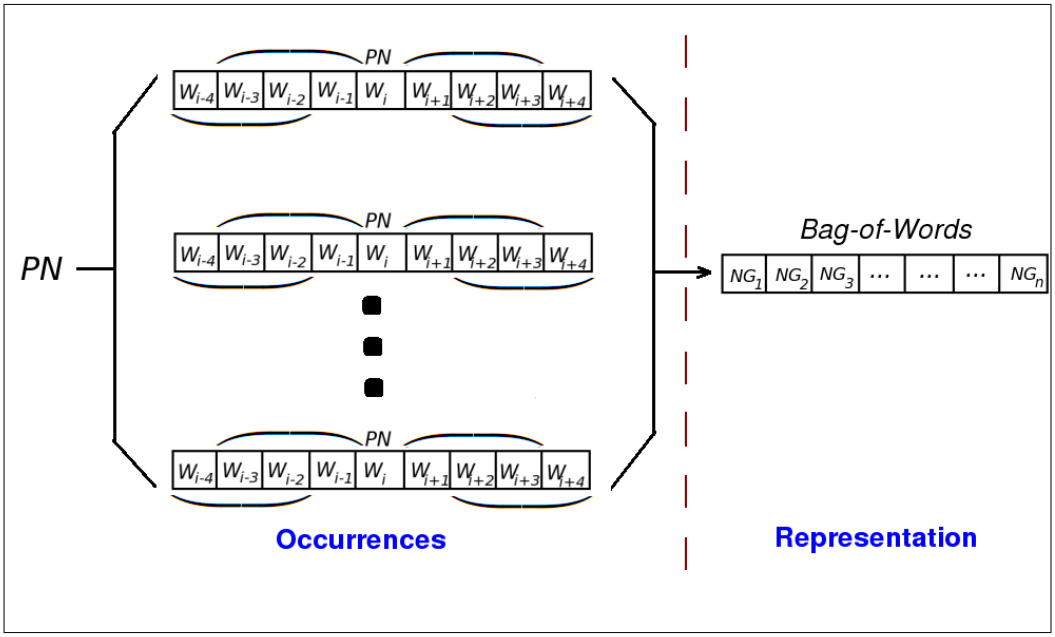

Figure 10.1: Bag-of-Words n-grams for PNs

10.3.3 Bag-of-Vectors (BoVs)

In this model, each PN in the text is represented with a bag of vectors in which each vector is a standard BoW for each occurrence of the PN (see Figure 10.2). Let's consider a PN P_1; its BoV representation is defined in Equation 10.2.

$$BoV(P_1) = \{b_{11}, b_{12} \dots b_{1i} \dots b_{1r}\} \tag{10.2}$$

when r is the number of occurrences of P_1 in the corpus and b_{1i} is a vector representing the *ith* occurrence of P_1 (as a BoW) in the corpus.

10.4 Similarity Functions and Clustering

This section is divided into two parts. First, we detail the similarity functions designed to handle the representation schemes presented in the previous section. Then, we present the clustering algorithm making the most of these similarities to build the PN clusters.

10.4.1 Similarity Functions

Many different similarity (or distance) functions can be used with a usual vectorial representation (that is, in our case the BoW representation). In this research,

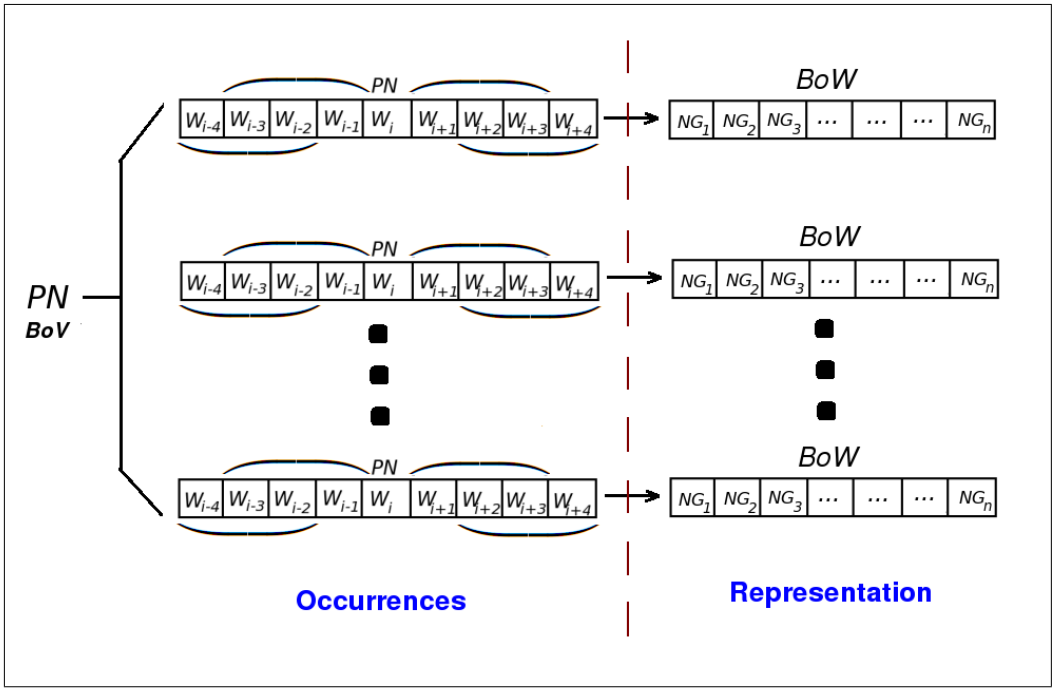

Figure 10.2: Bag-of-Vectors n-grams for PNs

we use three classic similarity functions: cosine, Jaccard and the Scalar Product [Manning and Schütze, 1999]. In addition to these usual similarity functions, we also propose the Power Scalar Product as detailed in Equation 10.3. Let us consider X and Y two vectors (BoWs), the Power Scalar Product is defined as:

$$\text{Power-Scalar}(X,Y) = \left(\sum_{i=1}^{n}(x_i * y_i)^p\right)^{1/p} \qquad (10.3)$$
$$X = \{x_1, x_2, ..., x_n\}, Y = \{y_1, y_2, ..., y_n\}$$

The intuition behind this new similarity function is to have a discriminative scalar product by increasing the parameter p. Indeed, increasing p will make the similarity of two vectors equal to the maximum value of $x_i * y_i$. In other words, this similarity function with larger p will be equal to the most important common feature between X and Y. Obviously, Equation 10.3 is the same as the Scalar Product when $p = 1$.

In order to use those usual similarity functions with BoVs, one needs to adapt them. The simplest strategy is to define a way to aggregate all the similarities computed from all the possible combination of vector comparison from the two BoVs considered using usual similarity functions. For instance, based on the work of Gosselin et al. [2007], one can define the similarity between two PNs based on

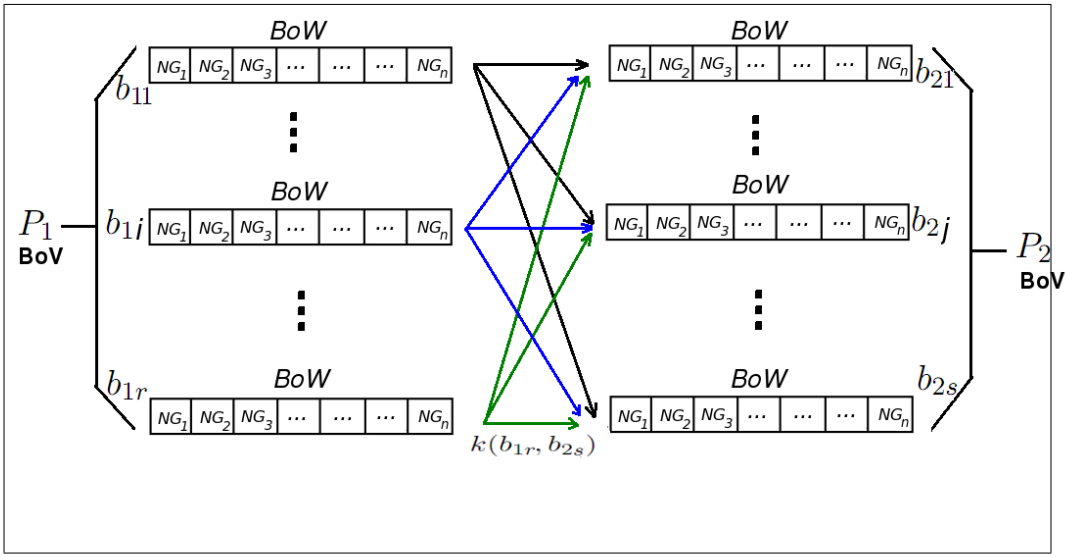

Figure 10.3: Similarity function on BoVs

their BoVs as the sum of the similarities among all the BoWs of both PN (see Figure 10.3).

Of course, many different ways can be used to define the general similarity function, such as sum-of-max or sum-of-sum of similarity. In this research, we use both sum-of-sum and sum-of-max definitions which are formulated respectively in Equation 10.4 and 10.5 where $P_1 = \{b_{11}, b_{12} \dots b_{1i} \dots b_{1r}\}$ and b_{1i} is a BoW of P_1 and $P_2 = \{b_{21}, b_{22} \dots b_{2j} \dots b_{2s}\}$ and b_{2j} is a BoW of P_2. In Equation 10.4, k could be any standard similarity function; r and s are the number of BoWs contained in P_1's and P_2's BoV respectively.

$$Sim_{SS}(P_1, P_2) = \sum_{i=1}^{r} \sum_{j=1}^{s} k(b_{1i}, b_{2j}) \tag{10.4}$$

$$Sim_{SM}(P_1, P_2) = \sum_{i=1}^{r} \max_{j=1}^{s} k(b_{1i}, b_{2j}) \tag{10.5}$$

However, both equations 10.4 and 10.5 have problems with more frequent objects. A frequent object has less sparse vector with higher weighted values; then, it has more chance to be similar to other objects. To solve this problem, Lyu [2005] proposed a normalized function which is formulated in Equation 10.6. In this definition, the final similarity is the average of all similarities. Assuming

that all instances of an object do not have the same importance, we rather need to have a discriminative similarity function.

$$Sim(P_1, P_2) = \frac{1}{|P_1|}\frac{1}{|P_2|}\sum_{i=1}^{|P_1|}\sum_{j=1}^{|P_2|} k(b_{1i}, b_{2j}) \tag{10.6}$$

In order to have such a discriminative kernel for calculating the similarity between two sets of bags, Lyu [2005] thus proposed a modified kernel on Bag-of-Bags (Equation 10.7).

$$Sim(P_1, P_2) = \frac{1}{|P_1|}\frac{1}{|P_2|}\sum_{i=1}^{|P_1|}\sum_{j=1}^{|P_2|} [k(b_{1i}, b_{2j})]^q \tag{10.7}$$

But this kernel can have very high discriminative value for higher q. In other words, the final similarity can goes infinite. This problem is solved in the next proposed Power Kernel (see below).

The complexity of computing these similarity functions for BoVs is higher than the standard BoWs since the final similarity is an aggregation of similarity between instances of pair of objects. In Equations 10.4 and 10.5, the complexity depends on r and s, i.e. the numbers of instances of the first and the second PN. In addition, the complexity of $k(b_{1i}, b_{2j})$ has to be considered. For both equations, computational cost is $O(r * s * n)$, where n is the size of the feature vector. But this complexity remains very low since each BoW is very sparse (even sparser than the unique BoW that is used in the standard representation). Indeed, for sparse data the computational cost of $k(b_{1i}, b_{2j})$ only depends on non-zero components of the vector for cosine, Jaccard and Power Scalar similarity functions.

Power kernel

Extending this idea in a Support Vector Machine context, Gosselin et al. [2007] proposed the so-called Power Kernel in order to increase the higher values and decrease lower values of partial similarities. This SVM kernel can of course be considered as a similarity function; we experiment this adapted similarity function defined in Equation 10.8, in order to build a discriminative similarity function.

In addition to Gosselin et al. Power Kernel, we define a new Power Kernel based on the sum of max of similarity as defined in Equation 10.9. Note that when $q = 1$, Equation 10.8 and Equation 10.9 are equivalent to Equation 10.4 and Equation 10.5 respectively.

$$Sim_{SSPK}(P_1, P_2) = \left(\sum_{i=1}^{r}\sum_{j=1}^{s} k(b_{1i}, b_{2j})^q\right)^{1/q} \tag{10.8}$$

$$Sim_{SMPK}(P_1, P_2) = \left(\sum_{i=1}^{r} \max_{j=1}^{s} k(b_{1i}, b_{2j})^q \right)^{1/q} \tag{10.9}$$

10.4.2 Markov Clustering

Generally, clustering is the (unsupervised) task of assigning a set of objects into groups called clusters so that the objects within the same cluster are more similar to each other than to the objects in any other cluster. In our case, our PN clustering task can be seen as a graph clustering in which each node in the graph is a PN and an edge is a relation between two PNs. In practice, this relation is defined as the similarity between the PNs, based on the common contextual features of their occurrences.

Among all the possible clustering algorithm, we decided to use Markov Clustering Algorithm (MCL) (see Section 7.2.2.1) which was proposed as a graph clustering algorithm [van Dongen, 2000a] and thus seems suited for our problem. It also offers an interesting advantage over more classic algorithms like k-means or k-medoid: MCL does not require the user to specify the expected number of clusters.

For our experiments, we used a Perl implementation of MCL called *minimcl* obtained from *http://micans.org/mcl*. The Inflation rate for this algorithm was set to 1.5 as suggested by the MCL developer in the source code.

10.5 Experiments

The previously defined representations and similarity functions with the Markov Clustering Algorithm (MCL) are used to cluster PNs in football reports. In this section, we first explain the evaluation metrics used, the experimental data. Then the results with the different similarity functions are presented and discussed.

10.5.1 Evaluation Metrics

As it has been previously said, the goal of the clustering is to have high intra-cluster similarity (similar objects in the same cluster) and low inter-clusters similarity (objects from different clusters are dissimilar); this is called an internal criterion. But having a good score for this internal criterion does not mean necessarily a good effectiveness. Evaluating clustering is thus mainly made with an external criterion [Manning et al., 2008], that is, using a ground-truth to find out how much the clustering results are similar to it.

Minute	Report
80	**Zigic** donne quelques frayeurs à **Gallas** et consorts en contrôlant un ballon chaud à gauche des 16 mètres au devant du Gunner. Le Valencian se trompe dans son contrôle et la France peut souffler.
82	Changement opéré par Raymond Domenech avec l'entrée d'**Alou Diarra** à la place de **Sidney Govou**,pour les dernières minutes. Une manière de colmater les brêches actuelles ?

Table 10.3: Minute-by-minute football report in French

This evaluation thus relies on the comparison of a ground-truth clustering and the clustering produced by the algorithm. Different metrics of cluster evaluation (or comparison) such as Purity or Rand Index [Rand, 1971] have been proposed in the literature which were explained in Section 7.2.3.2. Yet, these metrics are known to be not very discriminative, sometimes being over-optimistic, especially when the number of members in each cluster is relatively small [Vinh et al., 2010]. To the contrary, Adjusted Rand Index (ARI) is known to be robust as it is an adjusted-for-chance form of the Rand index. It is thus chosen as the main evaluation metric in this chapter.

10.5.2 Data

In this experiment, we use the same data set as in Chapter 9, called minute-by-minute reports which were extracted from French specialized websites. Let us remind that in this text, almost each minute of the football match is summarized with the description of the important events during that minute, including player replacement, fouls or goals (see Table 10.3).

For the experiments reported below, 4 football matches were considered; it corresponds to 819 sentences, 12155 words and 1163 occurrences of PNs (235 unique PNs). As explained in Chapter 9, the data has been annotated by experts [Fort and Claveau, 2012]. This human annotation resulted in a ground-truth composed of nine classes of PNs, including player names, coach names, etc., which are listed in Table 10.4. Unsurprisingly, the most frequent PNs in the

Cluster label	**N**	**Of total**
player	712	68%
team	114	11%
town	62	6%
trainer	44	4%
other	43	4%
country	26	2%
championship	26	2%
stadium	13	1%
referee	11	1%

Table 10.4: NE classes in ground-truth

Similarity	**BoW**	BoV_{SS}	BoV_{SSPK}
Cosine	8.46	47.88	40.13
Jaccard	9.95	48.19	30.33
Scalar Product	54.75	66.62	60.43
Power Scalar	8.46	64.77	71.27

Table 10.5: Similarity functions comparison with sum-of-sum, in terms of ARI (%)

reports are player names, which could make this class important to our model. It is also interesting to see how unbalanced these ground-truth clusters are.

10.5.3 Results

In this experiment, we evaluate three different models for PN clustering: Bag-of-Words, Bag-of-Vectors and a combination of BoV with Power Kernel. For all the models, we use the cosine, Jaccard, Scalar Product and Power Scalar similarity functions, and with all three models, we utilize Markov Clustering Algorithm. For these four similarity functions, we report the results for classic BoWs. For the BoV representation, the similarity measures for each vector can be combined with sum-of-sum and sum-of-max functions, or with the function that we called power kernel. In addition to this, we also perform a random clustering of the PN to serve as a baseline. All the results are presented in Tables 10.5 and 10.6.

The main result which is worth noting is that BoV outperforms BoW in every case. The new representation scheme, Bag-of-Vectors, thus seems more suited for this type of tasks. For the sum-of-sum model (Equation 10.8), the maximum

Similarity	**BoW**	BoV_{SM}	BoV_{SMPK}
Cosine	8.46	63.08	42.23
Jaccard	9.95	50.47	30.33
Scalar Product	54.75	64.63	57.05
Power Scalar	8.46	54.9	60.88

Table 10.6: Similarity functions comparison for sum-of-max on BoV, in ARI (%)

ARI is obtained with Power Scalar ($p = 5$) when combined with Power Kernel ($q = 3$; others q give slightly inferior but comparable results). As expected with the definition of ARI, random clustering yields 0. Even, the standard approaches with BoW are hardly above random clustering results. For example, cosine with BoW could not achieve better than 8.46% for ARI which could be considered as another base line system.

In addition to the sum-of-sum similarity function, we also examine the sum-of-max (Equation 10.9). The results are listed in Table 10.6. The comparison of the two tables show that sum-of-sum similarity made slightly better clusters with different similarity functions except for cosine. But, here again, these results are still far better than the usual BoW ones. Similarly to the sum-of-sum similarity function, Power Kernel does not improve the results for cosine, Jaccard and Scalar Product (whatever the factor q) but improves the result for Power Scalar.

Since the majority of Proper Nouns belong to the player names class, comparing the clustering results with the case in which all Proper Nouns are considered as player names is useful and gives us a base-line result. In this case, the ARI metric shows 42.47% which is better than cosine and Jaccard with different data descriptions. Yet, Scalar Product and Power Scalar are better than this base-line system both with BoV and BoV with Power Kernel.

10.5.4 Error Analysis

BoV with n-gram features appears to be a good model for clustering entities, obtaining better results compared to BoW model. In the final results, the number of clusters was an important parameter that can explain why some models works better than others. It is however interesting to have a closer look at the causes of errors in the final clustering results and study the PN classes in ground-truth that cannot be clustered with our model and why. To do so, we examine the errors for each class of the ground-truth.

We first calculate the precision and recall for each PN in the clusters based on the definition of B-Cubed precision and recall [Bagga and Baldwin, 1998]. The precision is expressed in Equation 10.10, in which PN_i is i^{th} PN in cluster C_j and

Class	Precision	Recall	F-Measure
player	88.60	91.47	90.01
referee	80.00	100.00	88.89
trainer	40.31	42.86	41.54
championship	25.00	100.00	40.00
town	55.42	22.31	31.82
team	18.14	30.61	22.78
other	15.08	25.00	18.82
country	10.43	37.50	16.32
stadium	7.67	50.00	13.30

Table 10.7: Class average precision for best model

$L(PN_i)$ denotes the class of PN_i.

$$Pre(PN_i, C_j) = \frac{|L(PN_i) \cap C_j|}{|C_j|} \tag{10.10}$$

Then, the average precision for each class in the ground-truth is computed, that is, the average precision of its members (Equation 10.11).

$$Pre_{ave}(C_j) = \sum_{i=1}^{N} w_i * Pre(PN_i, C_j) \tag{10.11}$$

In Equation 10.11, w_i is the weight of the i^{th} object in the cluster and N is the number of objects in the cluster. In this experiment, we consider all PNs in a cluster with the same weight: $1/N$. Similarly, recall (Rec_{ave}) can be calculated based on the idea of the B-Cubed scoring algorithm. We also calculate f-measure according to Equation 10.12.

$$\text{f-measure} = 2.\frac{Pre_{ave} * Rec_{ave}}{Pre_{ave} + Rec_{ave}} \tag{10.12}$$

An example of calculating precision and recall based on this method is depicted in Figure 10.4. In this example, for the entity e in the left side, there are four of five other similar entities in the same cluster which give us the precision based on the Equation 10.10 equal to $Pre(e, C_j) = \frac{4}{5}$. Similarly, in the right hand side, for the nominated entity e, the clustering algorithm grouped 4 of 6 similar entities in one cluster and 2 of the same entities in another cluster; so the recall is $Rec(e, C_j) = \frac{4}{6}$.

For our best model (Power Scalar similarity combined with sum-of-sum Power Kernel), the precision, recall and f-measure are reported in Table 10.7. The best

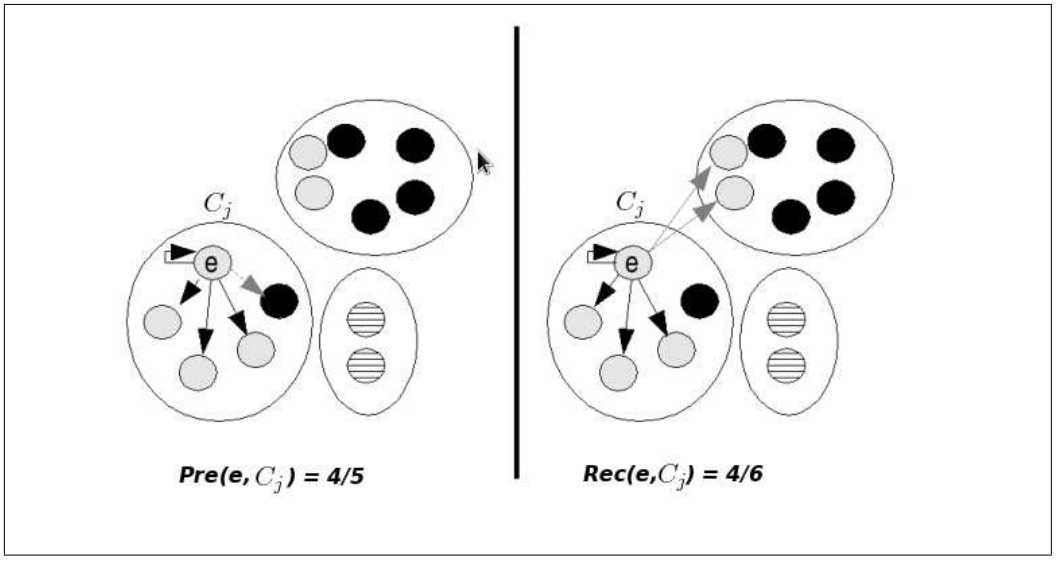

Figure 10.4: B-Cubed precision and recall [Bagga and Baldwin, 1998]

f-measure is for the player name class which is also the most important class in the reports. The most confused class with player name is *town*. The reason is that in some sentences, player names often come with a city name, making their contextual features very similar, and thus increasing the similarity between city names and player names.

The second best class is "referee" with a recall of 100% which means that all PNs in this class are in the same cluster. This high recall shows that n-grams occurring with "referee" PNs rarely come with other PNs in the report. For instance, a close examination of the corpus shows that referee names are almost always preceded by *Monsieur* (Eng. Sir), while other persons (players, trainers...) are not.

The class evaluation also shows that "stadium" is the most difficult class to cluster in this model. We found that n-grams around "stadium" PNs are spread out in the reports and, here again, near other PNs which makes the clustering difficult for this class because of low similarity between them.

It is also interesting to note that we use a simple PN detection technique solely based on the Part-of-Speech and it causes some errors. For example, "Guingampais" is guessed as a Proper Noun by TreeTagger (which does not have this word in its lexicon) but it is not a correct POS tag. Moreover, it also biases the n-grams counts and thus the idf used for the description of the other PNs. Conversely, no PN from the ground-truth is missing from the automatic clustering results. This simple detection system has thus a sufficiently good recall and decent precision for this application.

10.6 Conclusion and Future Work

In this research, reported in this chapter, we tackled an unsupervised Information Extraction problem for gathering entities and semantically group them with no apriori about their classes. In order to do so, we proposed a model for entity clustering based on the use of new representation schemes called Bag-of-Vectors. This representation keeps the effectiveness of the vectorial representation, and thus allows a fast and easy calculation of distances, while representing each occurrence of an entity independently. In order to compute these distances, we have shown that simple generalizations of usual vectorial similarity functions can be made. The whole approach, evaluated on a proper nouns clustering task in the football domain, outperformed standard approaches. In particular, the new Power-scalar similarity function that we proposed, combined with the Power-Kernel generalization allowed us to build a very discriminative model.

Let us remind of the goal of this research in which this thesis is done, that is, to find modules suitable for extracting information from multimedia documents (for instance, transcription of the speech they contain). So, we applied the proposed model on manually transcription of football matches but unfortunately in this preliminary experiments, the results were not that good. In the first step of error analysis, we found that the data were not uniformly transcribed and annotated. Moreover, we found more noise in the transcribed text compared to the minute-by-minute football report: misspelled PNs or some non-word tokens.

These results do not invalidate our approach, but highlight the need of minimal preprocessing to cover these transcription artifacts. Several other ideas, exploiting concepts proposed in this chapter, can be developed. The Bag-of-Vectors representation can, for example, be exploited in the context in information retrieval in which documents are often represented as Bag-of-Words. From a more fundamental point of view, many other similarity functions and many other ways to adapt them for BoV can be proposed. For instance, here we only used the maximum and the sum to aggregate the different vector similarities, and both can be seen as OR logical operators. Fuzzy logic offers many other logic operators to model the OR (T-conorms), and more generally many aggregation operators with well controlled properties that could be interesting to test in this context or more generally for information retrieval.

Chapter 11

Conclusion

In this chapter, we first review our work and highlight our main contributions. Then, we discuss the limitation of our work. In the final sub-section, some future work are proposed.

11.1 Summary and Contribution

In this thesis, we have detailed our research on building Information Extraction systems that are aimed to be used for multimedia indexing. Different researches have been done on extracting information from such documents (e.g. news broadcasting, sport videos) [Gravier et al., 2011; Buitelaar et al., 2008b; Reidsma et al., 2003] by using textual data. We assumed that some of these documents, such as news broadcasts, are not noisy because they mostly contains grammatically correct sentences with few or no unknown words. Since the goal of this thesis is robust IE systems, we decided to focus on more challenging documents (e.g. football match reports) which have more ungrammatical sentences in speech and more unknown words (proper nouns). Among the different textual sources of information related to the selected category of multimedia documents, we chose to use transcribed text and related external textual data available in the Web. From an application point of view, transcribed texts are also useful because of having temporal information in which each sentence is annotated with the exact time in the video. External textual data are helpful because of having less noise compared with transcribed text but without temporal information. However, in some domains (e.g. football), we can find some external textual sources with temporal information, although it is not as much detailed as transcribed text. We may have a summary for each minute of the report instead of having corresponding time interval for each sentence. These information with their temporal information (in second for transcribed text or minute for the second source of data) can be used to annotate multimedia documents. Then, these annotations

can be used in an Information Retrieval system to help users find answers for questions, for example, about important events in the documents.

One of the main objectives of this thesis was to be robust against noisy and small data. Our approach to reach this objective was to use simple and knowledge-light techniques as a guarantee of robustness that we assume to be mandatory for processing multimedia documents. Indeed, according to the property of our data, we decided not to use deep syntactic or semantic analysis of texts or sentences because the performance of these techniques decreases effectively for noisy data. More precisely, we used statistical analysis of text and some techniques inspired from Information Retrieval. More precisely, we utilize the idea of Language Modeling that has been used successfully in different Natural Language Processing applications (e.g. Automatic Speech Recognition, Machine Translation). We adapted this model according to the needs in this thesis. Moreover, we introduce a new data representation scheme for text processing which has been used successfully in image Information Retrieval domain.

In this thesis, we proposed three main modules of a robust Information Extraction system: Relation Extraction, Relation Discovery and Proper Noun Discovery (clustering). In the following, we remind the reader of our contributions in each task. Then, in the next sub-sections, issues worth explaining and foreseen improvements are proposed, based on a critical analysis of the inherent limitation of our work.

Relation Extraction by Shallow Linguistic Analysis: Relations between entities are one of the most important information in textual data and consequently in multimedia documents. In this task, we proposed a simple but effective supervised model to detect and classify relations between entities, assuming that all entities are annotated in the data. One of the most important part of any Relation Extraction system is its similarity function. Our contribution in this task was our proposed similarity function based on Language Modeling which is used with a kNN algorithm to classify relations. To calculate the similarity between two relations, we compute the probability of generating one relation form another one. In this calculation, we defined each relation as a set of n-grams in order to capture the word sequences. We demonstrated the effectiveness of this model compared with state-of-the-art models for Protein-Protein Interaction extraction task.

Relation Discovery model for noisy data: In some cases, we are interested to discover relation instead of recognition. So, Relation Extraction models cannot be useful anymore. This relation discovery task can be done by detecting and clustering them based on their similarities. The way of realizing this discovery task, and the function that we proposed to calculate the similarity between relations are our main contribution in this task.

Considering a relation as set of n-grams, we defined the similarity between two relations as the average conditional probability of having one relation when the other relation is given. In addition, in this task, we studied the importance of different unsupervised filters in order to separate (interesting, whatever their category) positive and negative (non interesting) relations.

A Bag-of-Vector Representation model for Proper Nouns Clustering: Entities are another important information in textual data. Most of the state-of-the-art systems on Named Entities are based on a supervised approach. But in our case, we defined this task to solve two main challenges in NE models: performing fine-grained NE detection without the need of annotated data. As a module of our robust Information Extraction model, we proposed a clustering model which does not need annotated data and can group proper nouns semantically in fine-grained clusters. In this model, our main contribution is the new instance based data representation. Defining each proper noun as one vector, based on contextual information of its occurrences in the text, is a popular vectorial data representation model, called Bag-of-Words (BoW). We assumed that each instance of a proper noun in the text is important, which is lost in BoW model. Thus, we proposed to use a representation, the Bag-of-Vectors (BoV), in which the occurrences are kept separated. We first adapted classical similarity functions (e.g. cosine, Jaccard) in order to work with BoV scheme. Moreover, in order to highlight the importance of each instance of proper nouns effectively, we proposed a discriminative version for each classical similarity function.

Although deep linguistic analysis has been used successfully for extracting information from text, we showed in this thesis that shallow linguistic analysis are effective in IE, specially when the data is noisy. Most of our contributions consisted in finding the suitable representation of the data rather than using complex Natural Language Processing or Machine Learning techniques. Besides the results we obtained with these models, there are some limitations to use them either to achieve higher performance or to use for larger data (scalability limitation). In the following, we review these limitations and explain some solutions.

11.2 Limitations

Of course, we do not claim to have solved all issues in this 3-year research. Hereafter, we review each model limitations and give some elements for possible solutions. Yet, seeking for complete solutions can be seen as a new research if the

limitation is critical in our model. Since our goal is to have robust system instead of scalable, almost all models have indeed scalability limitation.

Relation Extraction by Shallow Linguistic Analysis: The model that we proposed for Relation Extraction was based on a lazy learning classifier, k-Nearest Neighbors. For each new relation, its similarities to all other relations in the training data need to be computed. For huge data, this computation is too expensive. But if we cluster the training data, and represent each cluster with the centroid relation, each new relation needs only to be compared with the centroid of the cluster instead of all relations. In addition to the scalability problem, we may have difficulty to classify fine-grained relations with this model. In our model, we used unweighted n-grams for each relation to calculate the similarity which cannot distinguish important words for each relation. We only consider local features in our similarity function. Some n-grams are common among different relations and should have lower weight compare to those relations that co-occurred with the limited number of relations. We address this problem in our next model by considering weighted average conditional probability of having a relation if another relation has given. Similar approaches can be used with this model too.

Relation discovery model for noisy data: We used different techniques that have scalability problem in this model. For example, Markov Clustering algorithm complexity is $O(n^3)$ where n is the number of nodes in the graph (or the number of relations). Most of the MCL operation is matrix multiplication which can be reduced to $O(n^{2.807})$ [Strassen, 1986] or even less [Coppersmith and Winograd, 1990]. In this case, because of sparsity, it can be reduced to $O(\text{n } k^2)$ by using a sparse matrix implementation [van Dongen, 2000b]. The Language Modeling (n-gram) that we use is a bottleneck to use the model for larger data, however it can be replaced with a faster model similar to what Pauls and Klein [2011] have proposed. For the similarity function based on the average conditional probability, we consider equal weights for each n-gram in each relation. Indeed, each n-gram should have different weight by considering global context. As a solution, a tf-idf weight or similar can be used to take into account the global importance of each n-gram.

Bag-of-Vector representation scheme: The complexity of computing similarity functions for BoVs is higher than the standard BoWs since the final similarity is an aggregation of similarity between instances of pair of objects. In the case of having huge data with high frequency proper nouns, this similarity function may have problem, even with sparse vector model.

We have different solutions to reduce the number of features in the vector or the number of vectors for each proper noun. One solution for this problem is to represent each proper noun as a matrix by considering each vector of BoV as a row. Then different techniques (e.g. clustering, Singular Value Decomposition) can be used to reduce the number of rows or columns in the matrix. The clustering can be seen as combining similar occurrences of the proper noun together or as considering one vector for different but similar occurrence of a proper noun. Then, the final matrix can be decomposed to a new set of vectors to build a new BoV for each proper noun. For the clustering algorithm that is used in this model (MCL), we have already explained the limitations and suggest some solutions in the previous task. In this model we used a POS tagger in order to find proper nouns in the text. The results analysis showed that some errors originated from this process where some tokens are tagged as proper noun incorrectly.

Future Work
In the framework of Information Extraction from multimedia documents, we propose some topics as future researches in the following.

Fine-grained Information Extraction (relations and entities) is a challenging task for supervised and unsupervised models. In Chapter 8 we proposed a model based on kNN to classify relations. We also evaluated our model with fine-grained relations in bio-medical domain (see Table 8.8). We found that our model have some difficulties to classify relations when there are fine-grained. What we need is to re-classify relations that are classified by our kNN model which is reported as a solution to similar task in Visual Category Recognition [Zhang et al., 2006]. The same solution can be used for Relation Discovery when a model can discover positive and negative relations. Another discriminative model can discover fine-grained relations in each cluster.

Multimodal based approaches can help a model to integrate extracted information from different sources in one place. As we previously mentioned, different sources of textual data can be found for multimedia documents (for instance, minute-by-minute report for football matches). Integrating results from different sources can help to validate each extracted data and, even more interesting, finding contradictions in the extracted data. For example, in the football report, if the extracted score of the match in the transcribed text is different from the same information extracted from newspapers, we can remove the information from the first source assuming that extracting information from transcribed text has more errors compared to newspapers. Moreover, information from different sources can be complimentary

of each other and thus, integrating them helps to have more information for each period of time in the document. But, the most important challenge in this approach is the model of finding contradictions or complementary information and integrating them which is an interesting research topic.

Improving automatic transcribed text is the most important research that can be done for moving IE for multimedia documents forward. One of the early observation in this research was the poor results of an Automatic Speech Recognition (ASR) for generating transcribed text of football video reports. The main problem was caused by the high level of noise in stadium which always exists in such videos. Different techniques have been proposed in the literature to make an ASR system robust against such high level of noises (for example, [Al-Haddad et al., 2009]). Moreover, it is worth mentioning that one of the most frequent class of words in the football video reports is proper nouns. These proper nouns are unknown to the ASR system which can decrease the ASR performance effectively. Lecorvé et al. [2008] proposed to adapt the Language Model of the ASR system for specific domain to improve the performance which can be used in this field too. Indeed, they showed adapting the Language Modeling for keywords and nouns in the new domain can improve the ASR performance. In addition, a list of entity names (for instance, proper nouns) can be used in such adaptation. Our model in Chapter 10 can discover such entities from textual data and build a gazetteer list which can improve the ASR output.

Bibliography

Abney, S. (1991). Parsing by chunks. In *Principle-Based Parsing*, pages 257–278. Kluwer Academic Publishers.

Agichtein, E. and Ganti, V. (2004). Mining reference tables for automatic text segmentation. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 20–29, Seattle, WA, USA. ACM.

Airola, A., Pyysalo, S., Björne, J., Pahikkala, T., Ginter, F., and Salakoski, T. (2008). A graph kernel for protein-protein interaction extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 1–9, Columbus, Ohio, USA. ACL.

Al-Haddad, S. A., Samad, S. A., Hussain, A., Ishak, K. A., and Noor, A. O. (2009). Robust speech recognition using fusion techniques and adaptive filtering. *American Journal of Applied Sciences*, 6(2):290–295.

Amigo, E., Gonzalo, J., Artiles, J., and Verdejo, F. (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12:613–613.

Andersen, P. M., Hayes, P. J., Huettner, A. K., Schmandt, L. M., Nirenburg, I. B., and Weinstein, S. P. (1992). Automatic extraction of facts from press releases to generate news stories. In *Proceedings of the third conference on Applied natural language processing*, ANLC '92, pages 170–177, Trento, Italy. ACL.

Bagga, A. and Baldwin, B. (1998). Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 79–85, Montreal, Quebec, Canada. ACL.

Bahlmann, C., Haasdonk, B., and Burkhardt, H. (2002). On-line handwriting recognition with support vector machines - A kernel approach. In *proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition*, pages 49–54.

Baroni, M., Bernardini, S., Ferraresi, A., and Zanchetta, E. (2009). The WaCky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Bollegala, D. T., Matsuo, Y., and Ishizuka, M. (2010). Relational duality: Unsupervised extraction of semantic relations between entities on the web. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 151–160, Raleigh, North Carolina, USA. ACM.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Brill, E. (1992). A simple rule-based part of speech tagger. In *proceedings of the 3rd Applied Natural Language Processing Conference (ANLP)*, pages 152–155, Trento, Italy.

Buitelaar, P., Cimiano, P., Frank, A., Hartung, M., and Racioppa, S. (2008a). Ontology-based information extraction and integration from heterogeneous data sources. *International Journal of Human-Computer Studies*, 66(11):759 – 788.

Buitelaar, P., Declerck, T., Nemrava, J., and Sadlier, D. (2008b). Cross-media semantic indexing in the soccer domain. In *Proceedings of the 6th International Workshop on Content-Based Multimedia Indexing*, pages 296–301, London, UK. IEEE.

Bunescu, R. and Mooney, R. (2006). Subsequence kernels for relation extraction. *Advances in Neural Information Processing Systems*, 18:171–178.

Bunescu, R. C. and Mooney, R. J. (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 724–731, Vancouver, British Columbia, Canada. ACL.

Bunke, H. (2000). Recent developments in graph matching. In *Proceedings of 15th International Conference on Pattern Matching*, page 2117–2124, Barcelona.

Chakaravarthy, V. T., Gupta, H., Roy, P., and Mohania, M. (2006). Efficiently linking text documents with relevant structured information. In *Proceedings of the 32nd international conference on Very large data bases*, VLDB '06, pages 667–678, Seoul, Korea. VLDB Endowment.

Chang, C. C. and Lin, C. J. (2001). *LIBSVM: A library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chapelle, O., Vapnik, V., Bousquet, O., and Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159.

Chen, H., Benson, E., Naseem, T., and Barzilay, R. (2011). In-domain relation discovery with meta-constraints via posterior regularization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 530–540, Portland, Oregon. ACL.

Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Santa Cruz, California. ACL.

Chieu, H. L. and Ng, H. T. (2002). Named entity recognition: A maximum entropy approach using global information. In *Proceedings of the 19th international conference on Computational linguistics - Volume 1*, COLING '02, pages 1–7, Taipei, Taiwan. ACL.

Choi, M., Kim, H., and Croft, B. W. (2012). Dependency trigram model for social relation extraction from news articles. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, pages 1047–1048, Portland, Oregon, USA. ACM.

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 2:113–124.

Chomsky, N. (2002). *Syntactic Structures*. Mouton classic. Mouton De Gruyter.

Clear, J. H. (1993). The British national corpus. In *The digital word*, pages 163–187. MIT Press, Cambridge, MA, USA.

Collins, M. and Brooks, J. (1995). Prepositional phrase attachment through a backed-off model. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 27–38, Cambridge , Massachusetts, US.

Collins, M. and Singer, Y. (1999). Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, page 100–111, Maryland, USA.

Coppersmith, D. and Winograd, S. (1990). Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20:273–297.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21 –27.

Covington, M. A. (2001). A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102, Athens, Georgia, USA.

Cullingford, R. E. (1978). *Script Application: Computer Understanding of Newspaper Stories*. PhD thesis, Yale University.

Culotta, A. and Sorensen, J. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Barcelona, Spain. ACL.

Daelemans, W., van den Bosch, A., and Weijters, T. (1997). IGTree: Using trees for compression and classification in lazy learning algorithms. *Artificial Intelligence Review*, 11(1-5):407–423.

Debole, F. and Sebastiani, F. (2003). Supervised term weighting for automated text categorization. In *Proceedings of SAC-03, 18th ACM Symposium on Applied Computing*, pages 784–788, Melbourne, Florida, USA. ACM Press.

Dong, X. and Halevy, A. Y. (2005). A platform for personal information management and integration. In *Proceedings of the Second Biennial Conference on Innovative Data Systems Reasearch (CIDR)*, pages 119–130, Asilomar, CA, USA.

Doorenbos, R. B., Etzioni, O., and Weld, D. S. (1997). A scalable comparison-shopping agent for the World-Wide Web. In *Proceedings of the First International Conference on Autonomous Agents*, pages 39–48, Marina del Rey, CA, USA. ACM Press.

Ebadat, A. R. (2011). Extracting protein-protein interactions with language modelling. In *Proceedings of the Second Student Research Workshop associated with RANLP 2011*, pages 60–66, Hissar, Bulgaria. RANLP 2011 Organising Committee.

Ebadat, A. R., Claveau, V., and Sébillot, P. (2012). Proper noun semantic clustering using bag-of-vectors. In *Proceedings of the Twenty-Fifth International Florida Artificial Intelligence Research Society Conference*, Marco Island, Floride, USA. AAAI Press.

Ekbal, A., Sourjikova, E., Frank, A., and Ponzetto, S. P. (2010). Assessing the challenge of fine-grained named entity recognition and classification. In *Proceedings of the 2010 Named Entities Workshop*, NEWS '10, pages 93–101, Uppsala, Sweden. ACL.

Elagouni, K., Garcia, C., Mamalet, F., and Sebillot, P. (2012). Combining multi-scale character recognition and linguistic knowledge for natural scene text OCR. *Document Analysis Systems, IAPR International Workshop on*, 0:120–124.

Embley, D., Hurst, M., Lopresti, D., and Nagy, G. (2006). Table-processing paradigms: A research survey. *International Journal on Document Analysis and Recognition*, 8(2):66–86.

Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2004). Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 100–110, New York, NY, USA. ACM.

Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165:91–134.

Fayruzov, T., De Cock, M., Cornelis, C., and Hoste, V. (2008a). DEEPER: A full parsing based approach to protein relation extraction. In *Proceedings of the 6th European conference on Evolutionary computation, machine learning and data mining in bioinformatics*, EvoBIO'08, pages 36–47, Naples, Italy. Springer-Verlag.

Fayruzov, T., De Cock, M., Cornelis, C., and Hoste, V. (2008b). The role of syntactic features in protein interaction extraction. In *Proceedings of the 2nd international workshop on Data and text mining in bioinformatics*, DTMBIO '08, pages 61–68, Napa Valley, California, USA. ACM.

Fayruzov, T., De Cock, M., Cornelis, C., and Hoste, V. (2009). Linguistic feature analysis for protein interaction extraction. *BMC Bioinformatics*, 10.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Ann Arbor, Michigan. ACL.

Fleischman, M. and Hovy, E. (2002). Fine grained classification of named entities. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan. ACL.

Fort, K. and Claveau, V. (2012). Annotating football matches: Influence of the source medium on manual annotation. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Fundel, K., Kuffner, R., and Zimmer, R. (2007). RelEx: Relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371.

Gaifman, H. (1965). Dependency systems and phrase-structure systems. *Information and Control*, 8:304–337.

Gildea, D. and Jurafsky, D. (2001). Automatic labeling of semantic roles. *Computational Linguistics*, 28:245–288.

Giuliano, C., Lavelli, A., and Romano, L. (2006). Exploiting shallow linguistic information for relation extraction from biomedical literature. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006)*, pages 401–408, Trento, Italy.

Goadrich, M., Oliphant, L., and Shavlik, J. (2005). Learning to extract genic interactions using Gleaner. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 62–68, Bonn, Germany.

Gosselin, P., Cord, M., and Philipp-Foliguet, S. (2007). Kernels on bags of fuzzy regions for fast object retrieval. In *IEEE International Conference on Image Processing (ICI P07)*, volume 1, pages 177–180.

Gotoh, Y. and Renals, S. (2000). Information extraction from broadcast news. *Philosophical Transactions of the Royal Society of London, Series A*, 358:1295–1310.

Gravier, G., Guinaudeau, C., Lecorvé, G., and Sébillot, P. (2011). Exploiting speech for automatic TV delinearization: From streams to cross-media semantic navigation. *EURASIP Journal of Image and Video Processing*, 2011(0).

Greenwood, M. A., Stevenson, M., Guo, Y., Harkema, H., and Roberts, A. (2005). Automatically acquiring a linguistically motivated genic interaction extraction system. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 46–52, Bonn, Germany.

Grishman, R. and Sundheim, B. (1996). Message understanding conference-6: A brief history. In *Proceedings of the 16th conference on Computational linguistics - Volume 1*, COLING '96, pages 466–471, Copenhagen, Denmark. ACL.

Guinaudeau, C., Gravier, G., and Sébillot, P. (2009). Can automatic speech transcripts be used for large scale TV stream description and structuring? In *Proceedings of the 2009 11th IEEE International Symposium on Multimedia*, ISM '09, pages 489–494, San Diego, California, USA. IEEE Computer Society.

Hakenberg, J., Plake, C., Leser, U., Kirsch, H., and Rebholz-Schuhmann, D. (2005). LLL'05 challenge: Genic interaction extraction - Identification of language patterns based on alignement and finite state automata. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 38–45, Bonn, Germany.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1):10–18.

Harel, D. and Koren, Y. (2001). On clustering using random walks. In *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science*, volume 2245, chapter 3, pages 18–41. Springer Berlin / Heidelberg, Berlin, Heidelberg.

Hasegawa, T., Sekine, S., and Grishman, R. (2004). Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Barcelona, Spain. ACL.

Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92)*, pages 539–545, Nantes, France.

Hu, W., Xie, N., Li, L., Zeng, X., and Maybank, S. (2011). A survey on visual content-based video indexing and retrieval. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 41(6):797 –819.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2(1):193–218.

Isozaki, H. and Kazawa, H. (2002). Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th international conference on Computational linguistics - Volume 1*, COLING '02, pages 1–7, Taipei, Taiwan. ACL.

Jacobs, P. S. and Rau, L. F. (1990). SCISOR: Extracting information from on-line news. *Communication*, 33(11):88–97.

Joachims, T. (1998). Text categorization with suport vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, ECML '98, pages 137–142, Dorint-Parkhotel, Chemnitz, Germany. Springer-Verlag.

Jung, K. (2004). Text information extraction in images and video: A survey. *Pattern Recognition*, 37(5):977–997.

Katrenko, S., Scott Marshall, M., Roos, M., and Adriaans, P. (2005). Learning biological interactions from Medline abstracts. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 53–58, Bonn, Germany.

Kazama, J. and Torisawa, K. (2007). Exploiting Wikipedia as external knowledge for named entity recognition. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 698–707, Prague. ACL.

Kim, S., Yoon, J., Yang, J., and Park, S. (2010). Walk-weighted subsequence kernels for protein-protein interaction extraction. *BMC Bioinformatics*, 11:107.

Kim Sang, T., Erik, F., and De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147, Edmonton, Canada. ACL.

Kiryakov, A., Popov, B., Terziev, I., Manov, D., and Ognyanoff, D. (2004). Semantic annotation, indexing, and retrieval. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2(1):49 – 79.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, IJCAI'95, pages 1137–1143, Montreal, Quebec, Canada. Morgan Kaufmann Publishers Inc.

Kolbe, D., Zhu, Q., and Pramanik, S. (2010). Efficient k-nearest neighbor searching in nonordered discrete data spaces. *ACM Transactions on Information Systems*, 28(2):7:1–7:33.

Kondor, R. and Jebara, T. (2003). A kernel between sets of vectors. In *Proceedings of he twentieth International Conference on Machine Learning (ICML)*, Washington DC, USA. AAAI Press.

Kozareva, Z. (2006). Bootstrapping named entity recognition with automatically generated gazetteer lists. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 15–21, Trento, Italy. ACL.

Kubala, F., Schwartz, R., Stone, R., and Weischedel, R. (1998). Named entity extraction from speech. In *Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*, pages 287–292.

Kupiec, J. (1992). Robust part-of-speech tagging using a Hidden Markov Model. *Computer Speech and Language*, 6(3):225 – 242.

Kučera, H. and Francis, W. N. (1967). *Computational Analysis of Present-Day American English*. Brown University Press, Providence, RI.

Lawrence, S., Lee Giles, C., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE COMPUTER*, 32(6):67–71.

Lawto, J., Gauvain, J.-L., Lamel, L., Grefenstette, G., Gravier, G., Despres, J., Guinaudeau, C., and Sébillot, P. (2011). A scalable video search engine based on audio content indexing and topic segmentation. In *Proceedings of 2011 NEM Summit*, page 160 pages, Torino, Italy.

Lecorvé, G., Gravier, G., and Sébillot, P. (2008). On the use of web resources and natural language processing techniques to improve automatic speech recognition systems. In *proceedings of the Conference on Language Resources and Evaluation (LREC)*, Marrakech, Morocco. ELRA.

Lee, D., Jeong, O.-R., and Lee, S.-g. (2008). Opinion mining of customer feedback data on the web. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, ICUIMC '08, pages 230–235, Suwon, Korea. ACM.

Li, M.-H., Lin, L., Wang, X.-L., and Liu, T. (2007). Protein–protein interaction site prediction based on conditional random fields. *Bioinformatics*, 23(5):597–604.

Liao, W. and Veeramachaneni, S. (2009). A simple semi-supervised algorithm for named entity recognition. In *Proceedings of the NAACL HLT Workshop on Semi-supervised Learning for Natural Language Processing*, pages 58–65, Boulder, Colorad. ACL.

Liu, X., Zhang, S., Wei, F., and Zhou, M. (2011). Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 359–367, Portland, Oregon. ACL.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., and Watkins, C. (2002). Text classification using string kernels. *The Journal of Machine Learning Research*, 2:419–444.

Lytinen, S. L. and Gershman, A. (1986). ATRANS automatic processing of money transfer messages. In *Proceedings of the 5th National Conference on Artificial Intelligence (AAAI-86)*, pages 1089–1095, Trento, Italy. AAAI Press.

Lyu, S. (2005). Mercer kernels for object recognition with local features. In *IEEE Computer Vision and Pattern Recognition (CVPR 2005)*, pages 223–229. IEEE CONFERENCE PUBLICATIONS.

Manning, C., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Manning, C. D. and Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: The penn treebank. *Computational Linguistics - Special issue on using large corpora: II*, 19(2):313–330.

Marsh, E. and Perzanowski, D. (1998). MUC-7 evaluation of IE technology: Overview of results. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, Fairfax, Virginia.

McCallum, A. and Li, W. (2003a). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 188–191, Edmonton, Canada. ACL.

McCallum, A. and Li, W. (2003b). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons.

In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 188–191, Edmonton, Canada. ACL.

McCallum, A., Nigam, K., Rennie, J., and Seymore, K. (1999). A machine learning approach to building domain-specific search engines. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 662–667, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Medvés, L., Szilágyi, L., and Szilágyi, S. (2008). A modified Markov clustering approach for protein sequence clustering. In *Pattern Recognition in Bioinformatics*, volume 5265 of *Lecture Notes in Computer Science*, pages 110–120. Springer Berlin / Heidelberg.

Miyao, Y., Sagae, K., Sætre, R., Matsuzaki, T., and Tsujii, J. (2009). Evaluating contributions of natural language parsers to protein–protein interaction extraction. *Bioinformatics*, 25(3):394–400.

Momtazi, S., Lease, M., and Klakow, D. (2010). Effective term weighting for sentence retrieval. In *Proceedings of the 14th European conference on Research and advanced technology for digital libraries*, ECDL'10, pages 482–485, Glasgow, UK. Springer-Verlag.

Mondary, T. and Zargayouna, H. (2011). Quaero program, evaluation report: Results for task 3.3 on ontology acquisition, period 4. Technical report, LIPN.

Nadeau, D. and Satoshi, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30:3–26.

Nadeau, D., Turney, P., and Matwin, S. (2006). Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *Advances in Artificial Intelligence*, volume 4013 of *Lecture Notes in Computer Science*, pages 266–277. Springer Berlin / Heidelberg.

Ney, H., Essen, U., and Kneser, R. (1994). On structuring probabilistic dependencies in stochastic language modelling. *Computer Speech and Language*, 8:1–38.

Nédellec, C. (2005). Learning language in logic – Genic interaction extraction challenge. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 31–37, Bonn, Germany.

Panchenko, A. and Morozova, O. (2012). A study of hybrid similarity measures for semantic relation extraction. In *Proceedings of the Workshop on Innovative*

Hybrid Approaches to the Processing of Textual Data, pages 10–18, Avignon, France. ACL.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Pauls, A. and Klein, D. (2011). Faster and smaller n-gram language models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 258–267, Portland, Oregon. ACL.

Phuong, T. M., Lee, D., and Hyung Lee, K. (2003). Learning rules to extract protein interactions from biomedical text. In *Advanced in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science*, volume 2637, pages 148–158. Springer Verlag.

Pollard, C. and Sag, I. (1994). *Head-Driven Phrase Structure Grammar*. Chicago University Press, Chicago, Illinois.

Popelínský, L. and Blaťák, J. (2005). Learning genic interactions without expert domain knowledge: Comparison of different ILP algorithms. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 59–61, Bonn, Germany.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):pp. 846–850.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 147–155, Boulder, Colorado, USA. ACL.

Ratnaparkhi, A. (1996). A Maximum Entropy Model for Part-Of-Speech Tagging. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 133–142, Philadelphia, PA. USA.

Ravi, S., Knight, K., and Soricut, R. (2008). Automatic prediction of parser accuracy. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 887–896, Honolulu, Hawaii. ACL.

Reed, J. W., Jiao, Y., Potok, T. E., Klump, B. A., Elmore, M. T., and Hurson, A. R. (2006). TF-ICF: A new term weighting scheme for clustering dynamic data streams. In *Proceedings of the 5th International Conference on Machine Learning and Applications*, ICMLA '06, pages 258–263. IEEE Computer Society.

Reidsma, D., Kuper, J., Declerck, T., Saggion, H., and Cunningham, H. (2003). Cross-document annotation for multimedia retrieval. In *Proceedings 3rd Workshop on NLP and XML (NLPXML-2003)*, pages 41–48, Budapest, Hungary.

Riedel, S. and Klein, E. (2005). Genic interaction extraction with semantic and syntactic chains. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 69–74, Bonn, Germany.

Rink, B. and Harabagiu, S. (2011). A generative model for unsupervised discovery of relations and argument classes from clinical texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 519–528, Edinburgh, United Kingdom. ACL.

Rosenfeld, B. and Feldman, R. (2007). Clustering for unsupervised relation identification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 411–418, Lisbon, Portugal. ACM.

Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. (2008). LabelMe: A database and web-based tool for image annotation. *International Journal of Computer Vision (IJCV)*, 77(1-3):157–173.

Sætre, R., Sagae, K., and Tsujii, J. (2008). Syntactic features for protein-protein interaction extraction. In *Short Paper Proceedings of the 2nd International Symposium on Languages in Biology and Medicine (LBM 2007)*, volume ISSN 1613-0073319, pages 6.1–6.14, Singapore. CEUR Workshop Proceedings (CEUR-WS.org).

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

Santos, J. M. and Embrechts, M. (2009). On the use of the adjusted rand index as a metric for evaluating supervised classification. In *Proceedings of the 19th International Conference on Artificial Neural Networks: Part II*, ICANN '09, pages 175–184, Limassol, Cyprus. Springer-Verlag.

Sarawagi, S. (2008). Information extraction. *Foundations and Trends in Databases*, 1(3):261–377.

Sarmento, L., Carvalho, P., Silva, M. J., and de Oliveira, E. (2009). Automatic creation of a reference corpus for political opinion mining in user-generated content. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, TSA '09, pages 29–36, Hong Kong, China. ACM.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *proceedings of the International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK.

Shinyama, Y. and Sekine, S. (2006). Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL '06, pages 304–311, New York, New York. ACL.

Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *proceedings of the Ninth International Conference in Computer Vision (ICCV)*, ICCV '03, pages 1470–1477, Nice, France. IEEE Computer Society.

Snoek, C. G. M. and Worring, M. (2003). Time interval maximum entropy based event indexing in soccer video. In *Proceedings of the 2003 International Conference on Multimedia and Expo*, volume 3 of *ICME '03*, pages 481–484, Baltimore, Maryland, US. IEEE Computer Society.

Sobhana, N., Pabitra, M., and Ghosh, S. (2010). Conditional random field based named entity recognition in geological text. *International Journal of Computer Applications*.

Soucy, P. (2005). Beyond TFIDF weighting for text categorization in the vector space model. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005*, pages 1130–1135, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Stamou, G., Van, J., Pan, J. Z., Schreiber, G., and R. Smith, J. (2006). Multimedia annotations on the semantic web. *IEEE MultiMedia*, 13:86–90.

Stother, A. J. (2010). *On the Complexity of Matrix Multiplication*. PhD thesis, University of Edinburg.

Strassen, V. (1986). The asymptotic spectrum of tensors and the exponent of matrix multiplication. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, pages 49–54, Toronto, Canada.

Sturm, J., Kessens, J. M., Wester, M., de Wet, F., Sanders, E., and Strik, H. (2003). Automatic transcription of football commentaries in the MUMIS project. In *Proceedings of the 8th European Conference on Speech Communication and Technology, EUROSPEECH 2003 - INTERSPEECH 2003*, Geneva, Switzerland. ISCA.

Sun, C., Lin, L., Wang, X., and Guan, Y. (2007). Using maximum entropy model to extract protein-protein interaction information from biomedical literature. In *Proceedings of the intelligent computing 3rd international conference on Advanced intelligent computing theories and applications*, ICIC'07, pages 730–737, Qingdao, China. Springer-Verlag.

Takeuchi, K. and Collier, N. (2002). Use of support vector machines in extended named entity recognition. In *Proceedings of the 6th conference on Natural language learning*, volume 20 of *COLING-02*, pages 1–7, Taipei, Taiwan. ACL.

Tan, P.-N., Steinbach, M., and Kumar, V. (2006). *Introduction to Data Mining*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Toral, A. and Munoz, R. (2006). A proposal to automatically build and maintain gazetteers for named entity recognition by using Wikipedia. In *Proceedings of the EACL-2006 Workshop on NEW TEXT-Wikis and blogs and other dynamic text sources*, Trento, Italy.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, LIX:433–460.

van Dongen, S. (2000a). *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht.

van Dongen, S. (2000b). Performance criteria for graph clustering and Markov cluster experiments. Technical report, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam. Technical Report INS-R0012.

van Rijsbergen, C. J. (1974). Foundation of evaluation. *Documentation*, 30(4):365–373.

Verma, S., Vieweg, S., Corvey, W., Palen, L., Martin, J. H., Palmer, M., Schram, A., and Anderson, K. M. (2011). Natural language processing to the rescue? Extracting "situational awareness" Tweets during mass emergency. In *proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM)*, Barcelona, Spain. AAAI Press.

Vinh, N., Epps, J., and Bailey, J. (2010). Information theoretic measures for clusterings comparison. *The Journal of Machine Learning Research*.

Wang, C., Jing, F., Zhang, L., and Zhang, H.-J. (2006). Image annotation refinement using random walk with restarts. In *Proceedings of the 14th annual ACM international conference on Multimedia*, MULTIMEDIA 06, pages 647–650, Santa Barbara, CA, USA. ACM.

Wang, W., Besançon, R., Ferret, O., and Grau, B. (2011). Filtering and clustering relations for unsupervised information extraction in open domain. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 1405–1414, Glasgow, Scotland, UK. ACM.

Whitelaw, C., Kehlenbeck, A., Petrovic, N., and Ungar, L. (2008). Web-scale named entity recognition. In *proceedings of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 123–132, Napa Valley, California, USA. ACM.

Xiao, J., Su, J., Zhou, G., and Tan, C. (2005). Protein-protein interaction extraction: A supervised learning approach. In *proceedings of the First International Symposium on Semantic Mining in Biomedicine (SMBM)*, pages 51–59, Hinxton, Cambridgeshire, UK.

Yao, L., Haghighi, A., Riedel, S., and McCallum, A. (2011). Structured relation discovery using generative models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1456–1466, Edinburgh, United Kingdom. ACL.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 189–196, Cambridge, Massachusetts. ACL.

Zargayouna, H. (2010). Evaluation report: Results for task 3.3 on ontology acquisition period 3. Technical report, LIPN. Quaero project, task 3.3.

Zhang, H., Berg, A. C., Maire, M., and Malik, J. (2006). SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2126 – 2136.

Zhang, M., Su, J., Wang, D., Zhou, G., and Tan, C. L. (2005). Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. In *Proceedings of the Second international joint conference on Natural Language Processing*, IJCNLP'05, pages 378–389, Jeju Island, Korea. Springer-Verlag.

Zhao, Y. and Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis. Technical report, Department of Computer Science, University of Minnesota, Minneapolis, MN.

Zhou, G., Qian, L., and Fan, J. (2010). Tree kernel-based semantic relation extraction with rich syntactic and semantic information. *Information Sciences*, 180(8):1313–1325.

Zhou, G. and Su, J. (2002). Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 473–480, Philadelphia, Pennsylvania. ACL.

List of Figures

Résumé

Au cours de la dernière décennie, d'énormes quantités de documents multimédias ont été générées. Il est donc important de trouver un moyen de gérer ces données, pas tant d'un point de vue technique, mais plutôt du point de leur contenu. Chaque approche visant à faciliter ce processus nécessite donc d'avoir une connaissance approfondie du contenu de ces documents. Il existe deux familles d'approches pour obtenir un tel aperçu, soit par l'extraction d'informations à partir du document (par exemple, audio, image) ou en utilisant des données textuelles connexes extraites du document ou de sources externes (comme le Web). Notre travail se place dans cette seconde famille d'approches. Les informations extraites de ces textes peuvent ensuite être utilisées dans un cadre global et considérées comme des annotations des documents multimédias facilitant leur gestion.

L'un des principaux objectifs de cette thèse est donc de développer de tels systèmes d'extraction d'informations. Mais les textes extraits des documents multimédias étant en général petit et bruités, ce travail veille aussi à leur nécessaire robustesse. Notre approche pour atteindre cet objectif a donc été d'utiliser des techniques simples nécessitant peu de connaissances externes comme une garantie de robustesse. Pour ce faire, nous avons utilisé des techniques inspirées de la recherche d'information et de l'analyse statistique des textes.

Dans cette thèse, nous nous sommes concentré sur trois tâches : l'extraction supervisée de relations entre entités, la découverte de relations, et la découverte de classes d'entités. Dans la première tâche, l'extraction de relations, nous avons proposé une approche supervisée basée sur les modèles de langues et un algorithme d'apprentissage basé sur les instances, appelé k-plus-proches voisins. Les résultats expérimentaux ont montré l'efficacité de nos modèles qui utilisent des informations linguistiques simples à obtenir, contrairement aux systèmes de l'état de l'art qui utilisent des analyses linguistiques dites profondes, moins robustes. Dans la seconde tâche, nous sommes passé à un modèle non supervisé pour découvrir les relations au lieu d'en extraire des prédéfinies. Nous avons modélisé ce problème comme une tâche de clustering et défini une fonction de similarité là encore basée sur les modéles de langues. Les performances de ce modèle ont été évaluées sur un ensemble de textes décrivant le contenu de vidéos de matchs de football. Les résusltats obtenus ont montré l'intérêt de notre approche par rapport aux modèles classiques. En outre, nous avons étudié dans cette tâche l'importance de certains filtres indépendants du domaine. Enfin, dans la dernière tâche, nous nous sommes intéressés non plus aux relations mais aux entités. Celles-ci sont une source d'informations importante dans les documents et un élément nécessaire pour ensuite travailler les relations. Nous avons proposé une technique de clustering d'entités afin de découvrir et de faire émerger, sans a priori, des classes sémantiques parmi celles-ci. En particulier, nous avons pro-

posé une représentation nouvelle des données permettant de mieux tenir compte des chaque occurrence des entités. Ensuite, nous avons introduit une fonction de similarité discriminante afin de prendre en compte l'importance de chaque occurrence des noms propres dans le corpus.

En guise de conclusion, nous avons montré expérimentalement que des techniques simples, exigeant peu de connaissances a priori, et en utilisant des informations linguistique facilement accessibles peuvent être suffisantes pour extraire efficacement des informations précises à partir du texte. Dans notre cas, ces bons résultats ont été obtenus en choisissant une représentation adaptée pour les données, basée sur une analyse statistique ou des modèles de recherche d'information. Le chemin est encore long avant d'être en mesure de traiter directement des documents multimédia, mais nous espérons que nos propositions pourront servir de tremplin pour les recherches futures dans ce domaine.

Mots-clés Traitement automatique du langage naturel, Exploration de données, Recherche d'Information, Extraction d'Information, Analyse de regroupements, Bag-of-Vectors

Abstract

During the last decade, huge amounts of multimedia documents have been generated (for example, 72 hours of video are uploaded to YouTube each minute). It is therefore important to find a way to manage this data. Every approach to facilitate this process requires to have a deep understanding of the content of the documents. There are two main approaches to get such insights into the multimedia documents, either by extracting information from the document or by using related data from external sources (such as the Web). In the first approach, a variety of information can be extracted from the documents such as sequences of video shots, textual information and audio content. In the second approach, we have possibility to use external textual resources from the Web, to help the management of multimedia documents. In this thesis, we propose models to extract information from transcribed text and external textual resources. Then, these extracted information can be used in a global framework to be considered as annotations for multimedia documents in order to facilitate the management of such documents.

One of the main objectives of this thesis was to be robust against noisy and small data. Our approach to reach this objective was to use simple and knowledge-light techniques as a guarantee of robustness that we assume to be mandatory for processing multimedia documents. Indeed, according to the property of our data, we decided not to use deep syntactic or semantic analysis of texts or sentences because the performance of these techniques decreases effectively for noisy data. Instead, we used statistical analysis of text and some techniques inspired from Information Retrieval. More precisely, we utilized the idea of Language Modeling that has been used successfully in different Natural Language Processing applications (e.g. Automatic Speech Recognition, Machine Translation). We adapted this model according to the needs in this thesis. Moreover, we introduced a new data representation scheme for text processing which has been used successfully in image Information Retrieval domain.

In this thesis, we focused on three tasks: Relation Extraction, Relation Discovery and Proper noun clustering. In the first task, Relation Extraction, we proposed a supervised model based on a Language Modeling and an instance-based learning algorithm, called kNN. Experimental results showed the effectiveness of our models which use shallow linguistic information compared to state-of-the-art systems that use deep linguistic analysis. In the second task, we moved to unsupervised model to discover relations instead of extracting predefined ones. We modeled this problem as clustering task and defined a similarity function based on Language Modeling and average probability. The performance of this model was evaluated with a textual football reports, which showed improvements compare to classical model with cosine similarity function. Moreover, we studied the

importance of some domain independent filters in this task. Since each relation was between two entities, we defined the last task to cluster entities (more precisely, proper nouns) in order to discover and make emerge, without a priori, semantic classes.. In this task, we proposed to use a new data representation to keep each instance of the proper nouns separately. Then, we introduced a discriminative similarity function in order to take into account the importance of each occurrence of the proper nouns in the corpus.

As a conclusion, we experimentally showed that simple techniques, requiring few a priori knowledge, and using shallow linguistic information can be useful to effectively extract information from text. In our case, such results have indeed been achieved by choosing suited representation for the data, based on statistical analysis or Information Retrieval models. This is still a long road before being able to process raw multimedia documents, but we hope that these good results may now serve as a springboard for future researches in this field.

Keyword: Information Extraction, Semantic Clustering, Relation Extraction, Relation Discovery, Proper Noun Clustering, Similarity Function, Bag-of-Vectors, Language Modeling, N-gram